وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويـت

الذكاء الرقمي 9

الصـف التاسع

الفصل الدراسي الثاني



المرحلة المتوسطة

وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويـت



إشراف

أ. منى سالم عوض سالم (رئيس اللجنة)

تأليف

د. أشرف رضوان رضـــوان سليمـان
أ. منى مـــــرزوق مخـلد العـــازمـي

د. حسـام فتحـي سليمـــان وهبـــه
أ. منـار مصطفى عبدالحمـيد جمـال

أ. نوريـة حســـين علـــي بـوحمـــد
أ. أحمد فاروق عبد العظيم عبد الرحمن

د. عمرو محـــمود إبراهيم حبيب

تصميم

أ. ســـاره ياســـين عبدالله الأمـير
أ. أفـراح محـــمد هـدروس البذالـي

إخراج

أ. ســـاره ياســـين عبدالله الأمـير

الطبعة الأولى
1447 هـ
2026/2025 م

حقوق التأليف والطبع والنشر محفوظة لوزارة التربية - الإدارة العامة للتواجيه والبحوث والمناهج
إدارة البحوث التربوية والمناهج

الطبعة الأولى 2026/2025م

المراجعة العلمية

أ. علياء عباس حسن الصفــار

الفريق المساند :

أ. أحمد محمد عبدالله عيسى - تصميم

أ. سنية محمد علي المؤمـــن - تصميم

أودع بمكتبة الوزارة تحت رقم (7) بتاريخ 2026/1/12م


بسم الله الرحمن الرحيم



حضرة صاحب السمو الشيخ مشعل الأحمد الجابر الصباح

أمير دولة الكويت

H.H. Sheikh Meshal AL-Ahmad Al-Jaber Al-Sabah
Amir Of The State Of Kuwait



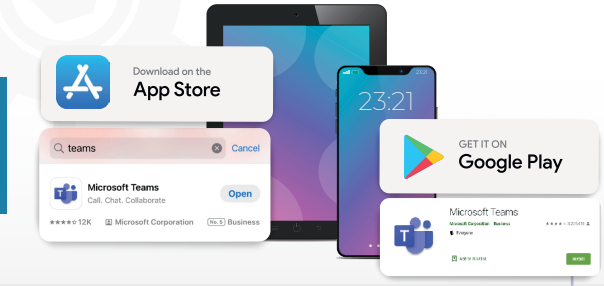
سمو الشيخ صباح خالد الحمد الصباح
ولي عهد دولة الكويت

H. H. Sheikh Sabah Khaled Al-Hamad Al-Sabah
Crown Prince Of The State Of Kuwait

الفصل الرقـــمي Digital Classroom

أولًا :تحميل وتثبيت تطبيق Microsoft Teams

الدخول على متجر تطبيقات الأجهزة الرقمية.



من الأجهزة الذكية



من جهاز الحاسوب

احرص على تحديث تطبيق Microsoft Teams بشكل دوري.

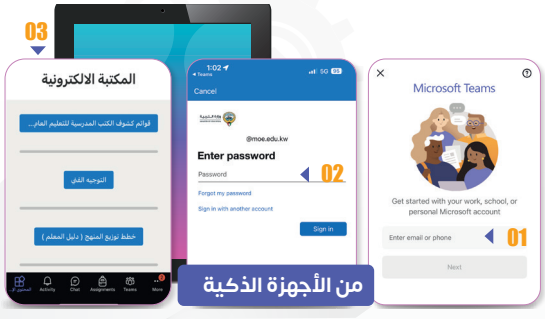
ثانياً: تفعيل Microsoft Teams على الجهاز الرقمي

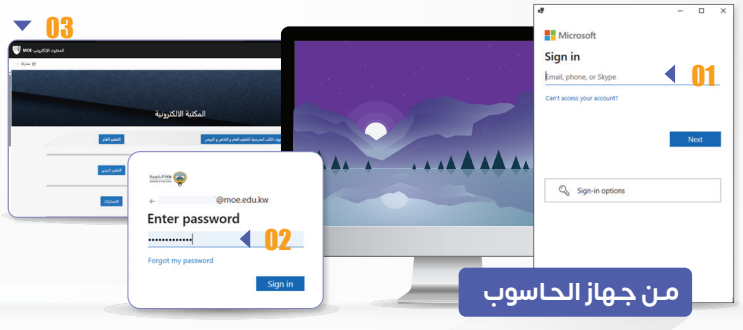
لتشغيل التطبيق اتبع الخطوات التالية:

01 أدخل اسم المستخدم: البريد الإلكتروني الرسمي الخاص بحساب Teams.

02 أدخل كلمة المرور الخاصة بحساب Teams.

03 تنقل بين واجهات التطبيق مثل المكتبة الإلكترونية وفرق المواد الدراسية.





احتفظ بكلمة المرور في مكان آمن لتسهيل تسجيل الدخول لاحقًا.

لا تشارك بياناتك مع أي شخص غير موثوق به.



مـن هـو حـمد ؟...

هـو مُساعد تعليمـي رقمـي ذكـي يعتمـد علـى تقنيـات الـذكاء الاصطناعـي، مُصمـم لتقديـم الدعـم التعليمـي التفاعلـي لكافـة المناهـج الدراسـية.

حمد دائمًا معك... لتتعلّم بثقة وتنجح بامتياز.

مع حـمد Chat

مـا هـي خـدمـة حـمـد شـات؟

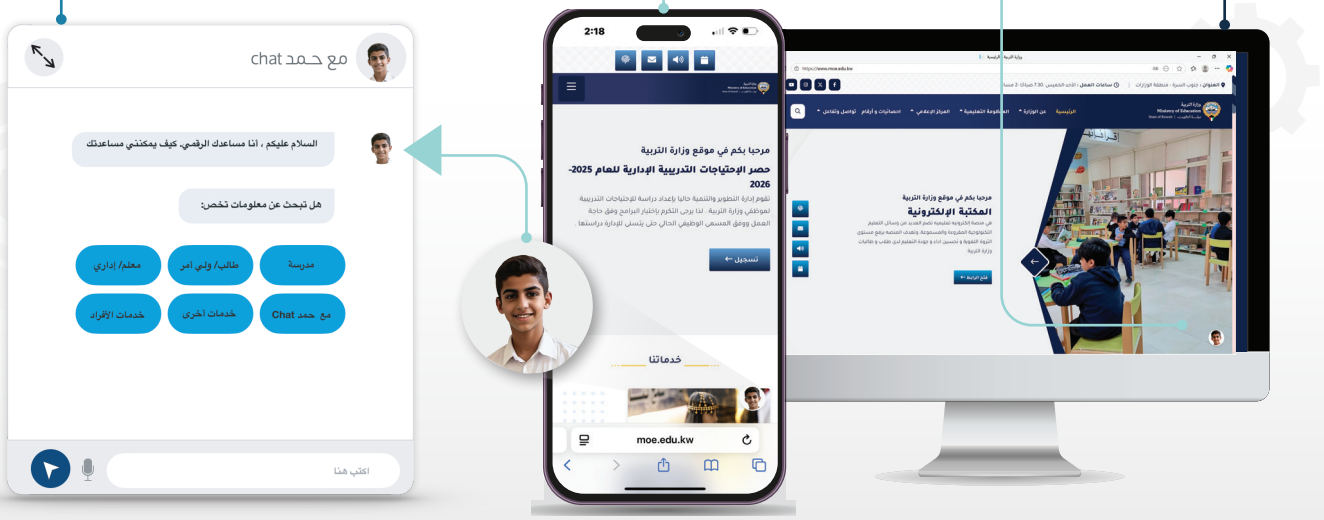
هـي خدمـة المحادثـة الذكيـة المقدمـة مـن وزارة التربيـة فـي دولـة الكويـت، تعتمـد علـى الـذكاء الاصطـناعي، تتمثـل وظيفتـه الأساسـية فـي تسـهيل عمليـة الفهـم والاسـتيعاب مـن خلال تقديـم الشـروحات المبسـطة، والإجابـة علـى الاستفسـارات، وتوجيـه المتعلميـن خلال مسـيرتهم التعليميـة.

أيـــن أجـــده؟

01
- زيارة الموقع الرسمي
www.moe.edu.Kw
- أو تطبيق وزارة التربية

02
الضغط عـلى
صورة حمد شات

03
اكتب استفساراتك، وستتلقى الإجابات بشكل فوري.



المواطنة الرقمية



يلتزم بالأمانة الفكرية.

يحترم الثقافات والمجتمعات في البيئة الافتراضية.

يحافظ على المعلومات الشخصية.

يحمي نفسه ضد المعتقدات غير السليمة التي تنتشر عبر الوسائط.

يدير الوقت الذي يقضيه في استخدام التكنولوجيا.

يحذر من التنمر الإلكتروني.

مِفتاح الكتاب

الاستكشاف



ربط المحتوى الدراسي بحياة المتعلم اليومية من خلال مواقف ومشكلات واقعية، تُقدّم بأسلوب مبتكر يعزز شعوره بأهمية ما يتعلمه

نواتج التعلم



ما يُتوقع أن يكتسبه المتعلم من معارف، أو يفهمه من مفاهيم، أو يؤديه من مهارات، أو يتبناه من قيم وسلوكيات بنهاية الدرس

التعلم



تقديم المحتوى العلمي بأسلوب مبسّط يناسب المرحلة العمرية للمتعلمين، ويُسهّل فهمهم للمعلومة واستيعابها

مصادر التعلم



استخدام رمز QR لتيسير وصول المتعلم إلى مصادر تعليمية إضافية تدعم فهمه للمادة العلمية

الأنشطة / المهام



مجموعة تدريبات توضح وتُعزّز مفهوم الدرس، مع تشجيع المتعلمين على تنفيذها وابتكار أفكار جديدة من خلالها.

عبّر عن -رأيك-



تدريب المتعلم على التقييم الذاتي لتعزيز قدرته على تحسين أدائه من خلال التعرف على نقاط القوة والضعف، مما يسهّل على ولي الأمر متابعة تقدّمه

التطبيق / المشروع



مجموعة ممارسات عملية ممتعة تعزز فهم المادة العلمية وتشجّع المتعلم على تطبيقها بطريقة تفاعلية

ملاحظات المعلم



ملاحظات يسجلها المعلم بعد متابعة أداء المتعلم، تعكس مدى فهمه لمفهوم الدرس وتحقيقه للكفاية المطلوبة.

في وقت فراغك



أنشطة منزلية يُنفذها المتعلم لتعزيز فهمه للمادة العلمية وترسيخ المفاهيم المكتسبة

ملاحظات ولي الأمر



تمكّن ولي الأمر من متابعة كتاب المتعلم وتدوين ملاحظاته وتعليقاته لدعم التعلم ومتابعة التقدّم

صحتك تهمنا



احرص على أداء بعض التمارين أثناء استخدامك جهاز الحاسوب



















حياة رقمية آمنة



اقض وقتاً أكبر مع أسرتك بعيداً
عن العالم الرقمي



حافظ على التمرينات الرياضية
بشكل دوري



احم أجهزتك الرقمية ببرامج
الحماية المناسبة



التزم بأخلاقيات التخاطب
والمحادثة مع الآخرين



استخدم كلمات مرور مركبة



تأكد من صيانة وتحديث أجهزتك
الرقمية بشكل مستمر



اكتسب معلومة وقدم فكرة كلما
استخدمت الأجهزة الرقمية



تأكد من عدم رفع ملفاتك
الشخصية على الإنترنت



لا تستخدم شبكات إنترنت غير
موثوقة



تخلص من الأجهزة الرقمية في الأماكن
الصحيحة التي تسمح بإعادة التدوير



تأكد من ترتيب التوصيلات الكهربائية
والأحمال المناسبة



لا تحاول الشراء من الإنترنت
بمفردك

المحـــتوى

العنوان	الصفحة
المقدمة	19
الوحدة الأولى: المعالجة الرقمية Digital Processing	
هندسة التطبيقات والذكاء البرمجي.	21
□ مدخل إلى تطبيقات الأجهزة الذكية.	23
– مفهوم تطوير تطبيقات الأجهزة الذكية.	26
□ بيئة MIT App Inventor.	28
– المكونات الرئيسية.	29
– واجهة المصمم Designer.	29
– واجهة الكتل البرمجية Blocks.	30
□ التعامل مع MIT App Inventor .	35
– إنشاء مشروع جديد.	37
□ واجهة الكتل والأحداث Blocks & Events Editor.	53
– كتل الأحداث.	55
– كتل الإجراءات.	57
□ الشاشات المتعددة Multiple Screens.	73
– مميزات تعدد الشاشات.	77
– إضافة شاشة جديدة للتطبيق.	79
□ الجملة الشرطية if (1).	89
– مفهوم ومكونات الجملة الشرطية.	91
□ الجملة الشرطية if (2).	101
– علامة «Blue Gear Sign».	104
– الشروط المركبة AND - OR .	105
□ التعامل مع النصوص والوسائط.	117
– مكون الصوت Sound.	119
– مكون المشغل Player.	120

المحـــتوى

العنوان	الصفحة
□ المتغيرات Variables (1).	133
– أنواع المتغيرات.	136
– إنشاء المتغيرات.	137
□ المتغيرات Variables (2).	149
– أنواع المتغيرات حسب البيانات.	151
– تعيين وتحديث قيمة المتغيرات.	152
□ إدارة التاريخ والزمن في التطبيق.	161
– مكوّن الساعة Clock.	163
– مكون التاريخ DatePicker.	164
– معالجة النصوص Join.	166
إدارة البيانات والعمليات الذكية	175
□ التعامل مع القوائم List (1).	177
– المكوّن ListPicker.	179
– الفهرس Index.	180
□ التعامل مع القوائم List (2).	191
– المكون ListPicker.	193
□ الحلقات التكرارية Loops.	207
– أنواع الحلقات التكرارية.	210
□ الإجراءات Procedures.	221
– إنشاء الإجراءات.	224
□ الدوال Functions.	235
– الدوال المدمجة.	238
□ قواعد البيانات TintDB (1).	243
– حفظ واسترجاع البيانات.	246

المحـــتوى

العنوان | الصفحة

- قواعد البيانات TintDB (2). 255
  - توظيف TinyDB مع القوائم. 257
  - استخدام ملفات CSV. 258
- قواعد البيانات TintDB (2). 265
  - حذف وتحديث البيانات. 237
- إنشاء شاشة رئيسية للتنقل بالتطبيق. 277
  - تنظيم الشاشة الرئيسية. 279
- أداة دمج التطبيقات AI2MergerAp. 285
  - دمج واجهة مستخدم موحدة في مشاريع متعددة. 287

الوحدة الثانية : المنتجات الرقمية Digital Products

وحدة المشروعات Projects Unit.

- فكرة وأهداف المشروع. 297
- مهارات المشروع. 298
- إجراءات تنفيذ المشروع. 299
- تطبيق Bright Moments In Kuwait's History. 301
- نماذج لأفكار مشروعات متنوعة. 303

المراجع 321

المقدمة

عزيزي المبتكر الرقمي... صانع الحلول الذكية،

يتميز عالمنا اليوم بتسارع رقمي كبير أصبحت فيه التطبيقات الذكية جزءًا أساسيًا من حياتنا اليومية. ومن هنا تبرز أهمية انتقالك من مجرد مستخدم للتقنية إلى منتج ومبتكر يمتلك المهارات التي تمكّنه من فهم كيفية عمل التطبيقات وتصميم حلول رقمية تخدمه وتخدم مجتمعه.

يأتي هذا الكتاب « الذكاء الرقمي » ليكون بوابتك العملية نحو بناء جيل واعٍ بالتكنولوجيا، قادر على توظيفها بصورة إيجابية وفعّالة، جيل يطمح للريادة ويسهم في تحقيق رؤية الكويت 2035 نحو مجتمع رقمي ذكي ومستدام.

ستخوض خلال هذا الفصل الدراسي رحلة تطبيقية شيّقة تتوزع على محاور أساسية تهدف إلى صقل مهاراتك البرمجية:

المحور الأول: هندسة التطبيقات والذكاء البرمجي

ستتعرّف في هذه الوحدة على الأساليب المتقدمة لبناء واجهات المستخدم UI - User Interface وتجربة الاستخدام UX - User Experience من خلال MIT App Inventor ، وهي بيئة تعليمية بصرية تعتمد على السحب والإفلات، وتبسّط المفاهيم البرمجية المعقدة، وتمكّنك من بناء تطبيقات تعمل على الأجهزة الذكية دون الحاجة إلى كتابة أكواد برمجية نصية، من خلال هذه التجربة ستكتشف أن البرمجة هي فن ترتيب الأفكار وتنظيمها قبل أن تكون مجرد أوامر يُنفّذها الحاسوب.

كما ستتعرّف على أساسيات التفكير الحاسوبي مثل: تحليل المشكلة، وتحديد عناصرها، وتنظيم الأفكار وربطها بمنطق برمجي واضح، وتصميم الحلول الرقمية بكفاءة، ثم اختبار التطبيق وتحسينه.

المقدمة

المحور الثاني: إدارة البيانات والعمليات الذكية

في هذا المحور ستخطو نحو الاحترافية من خلال تنفيذ مشاريع برمجية متكاملة، تكتسب فيها مهارات متقدمة تبدأ من تنظيم وتمثيل البيانات بفعالية باستخدام القوائم والفهرس والحلقات التكرارية، مروراً بتطبيق التحقق والمنطق البرمجي الذي يضمن دقة عمل التطبيق، وصولاً إلى احتراف إدارة قواعد البيانات TinyDB لحفظ واسترجاع المعلومات. كما ستتمكن من تفعيل دور الدوال والإجراءات لتبسيط العمليات البرمجية المعقدة، مع تصميم واجهة احترافية للشاشة الرئيسية لتكون نقطة الانطلاق المثالية لتطبيقك.

وفي الختام، ستجد في هذا الكتاب مجموعة من الأنشطة العملية وأوراق العمل التي تمثل ملف إنجازك الرقمي، حيث يُعد كل مشروع تنفذه خطوة جديدة في بناء مهاراتك المستقبلية.

نحن نثق بقدرتك على الإبداع، ونتطلع لرؤية ابتكاراتك التي ستسهم في رفعة وطننا الغالي.

رحلة ممتعة في عالم الابتكار الرقمي!

المؤلفون

هندسة التطبيقات والذكاء البرمجي

1. مدخل إلى تطبيقات الأجهزة الذكية.
2. بيئة MIT App Inventor.
3. طرق التعامل مع MIT App Inventor.
4. واجهة الكتل والأحداث Blocks & Events Editor.
5. الشاشات المتعددة Multiple Screens.
6. الجملة الشرطية if.
7. التعامل مع النصوص والوسائط.
8. المتغيرات Variables.
9. إدارة التاريخ في التطبيق .

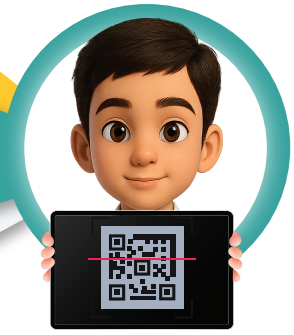
إدارة البيانات والعمليات الذكية

1. التعامل مع القوائم Lists .
2. الحلقات التكرارية Loops.
3. الإجراءات Procedures.
4. الدوال Functions.
5. قواعد البيانات TinyDB.
6. إنشاء شاشة رئيسية.
7. أداة دمج التطبيقات AI2MergerAp.

مدخل إلى تطوير تطبيقات الأجهزة الذكية من خلال منصة MIT App Inventor

نواتج التعلم

- التعرف على مفهوم تطوير تطبيقات الأجهزة الذكية.
- التعرف على دورة حياة تطوير البرمجيات SDLC.
- التمييز بين أنظمة التشغيل المختلفة للأجهزة الذكية.
- وصف المكونات الرئيسة لواجهة المصمم Designer.
- وصف المكونات الرئيسة لواجهة الكتل البرمجية Blocks.
- التمييز بين بيئة البرمجة النصية وبيئة البرمجة المرئية.
- توظيف مهارات التفكير المنطقي والتنظيم المكاني في تصميم الواجهة البرمجية.
- إظهار قيمة الإبداع والتجريب في تصميم واجهات التطبيقات التعليمية.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



كيف يمكن لمصمم التطبيق أن يرتب المكونات مثل (أزرار - صور - نصوص) على شاشة الهاتف بحيث تظهر بشكل منظم وسهل الاستخدام؟

لاحظ مكونات شاشة تطبيق هويتي واكمل ما يلي:

- **دوّن اسم عنصرين من عناصر شاشة التطبيق:**

1. .....
2. .....



شكل (1): واجهة تطبيق هويتي



التعلـــــم

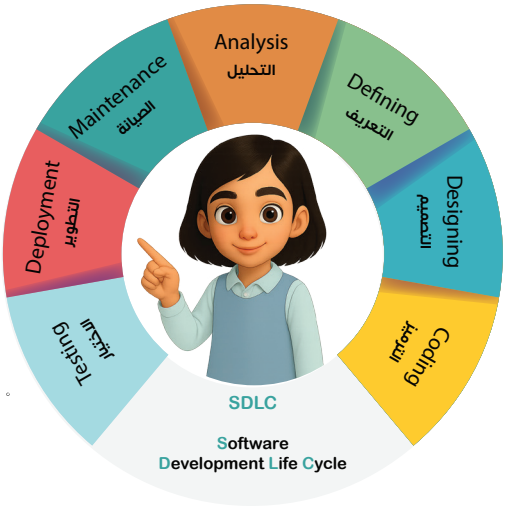
مدخل إلى تطوير تطبيقات الأجهزة الذكية:

مفهوم تطوير تطبيقات الأجهزة الذكية.

تصميم وبرمجة تطبيقات تعمل على الأجهزة الذكية باستخدام لغات وأدوات خاصة تتيح للمستخدم التفاعل مع الجهاز (مثل الكاميرا، الموقع، الإنترنت، إلخ).

دورة حياة تطوير البرمجيات SDLC Software Development Life Cycle:

تُعد دورة حياة تطوير البرمجيات خطة عمل منظمة تساعد المصممين والمطورين على إنتاج برامج ناجحة وفعالة تلبي احتياجات المستخدمين، وهي تشبه تمامًا مراحل بناء أي مشروع مثل بناء منزل، حيث تبدأ من الفكرة وتنتهي بتسليم المشروع جاهزًا.



شكل (2): مخطط يوضح دورة حياة تطوير البرمجيات.

مراحل دورة حياة تطوير البرمجيات:

1. **التخطيط Planning:**
   تحديد الأهداف العامة للتطبيق، تقدير الأهداف، وضع خطة عمل أولية.

2. **التحليل Analysis:**
   تحليل المتطلبات المحددة للتطبيق، وتحديد الموارد اللازمة لتلبية المتطلبات.

3. **التصميم Design:**
   تحديد المواصفات الهيكلية، بما في ذلك واجهات المستخدم User Interface - UI وتصميم قاعدة البيانات وتدفق البيانات.

4. **التطوير Development:**
   كتابة التعليمات البرمجية للمنتج بناءً على مواصفات التصميم.

5. **الاختبار Testing:**
   التحقق من أن التطبيق يعمل بشكل صحيح وخالٍ من الأخطاء، ويشمل اختبارات مثل اختبار الوحدة واختبار التكامل واختبار النظام.

6. **النشر Deployment:**
   نشر البرنامج ليصبح جاهزًا للاستخدام.

7. **الصيانة Maintenance:**
   توفير التحديثات والدعم المستمر بعد النشر ، ومعالجة أي مشكلات جديدة أو تعديلات مطلوبة بناءً على ملاحظات المستخدمين.

لاحظ

معظم التطبيقات الحالية تعتمد على تقنيات الذكاء الاصطناعي وهو أغلب ما يميز تطبيقات الأجهزة الذكية حالياً.



نشـــاط 1

- اذكر اثنان من تطبيقات الهاتف التي تتضمن تقنيات الذكاء الاصطناعي:

1. .....
2. .....

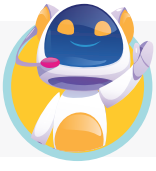
يمكن البحث من خلال مواقع الذكاء الاصطناعي التوليدي.

أنظمة تطبيقات الهواتف الذكية:

تعمل تطبيقات الهواتف الذكية من خلال أنظمة تشغيل مختلفة، وأشهر هذه الأنظمة الحالية:

HarmonyOS	Android	iOS
إنتاج شركة HUAWEI	إنتاج شركة Google	إنتاج شركة Apple

برنامج / منصة MIT App Inventor



هل فكرت يومًا كيف يتم إنشاء التطبيقات التي تستخدمها على هاتفك الذكي؟

يمكن تطوير تطبيقات الأجهزة الذكية من خلال العديد من المنصات ومنها منصة MIT App Inventor ، هي منصة تطوير مجانية تعمل على الإنترنت لبناء تطبيقات متكاملة للهواتف الذكية والأجهزة اللوحية تم تطويرها بواسطة شركة google، ثم أصبحت لاحقاً تحت إشراف معهد ماساتشوستس للتكنولوجيا Massachusetts Institute of Technology (MIT) ومن مميزاتها :

مجانية وسهلة الاستخدام.	لا تحتاج خبرة سابقة في البرمجة.	يمكن تجربة التطبيق مباشرة على الهاتف أو المحاكي.
يمكن أن تعمل على الجهاز بدون اتصال بالإنترنت.	مناسبة للمشاريع التعليمية والابتكارية.	تستخدم الكتل البرمجية بطريقة السحب والإفلات.



المكونات الرئيسية لبيئة عمل برنامج/منصة MIT App Inventor:

تتكون بيئة عمل البرنامج / المنصة من واجهتين أساسيتين :

- واجهة المصمم Designer :

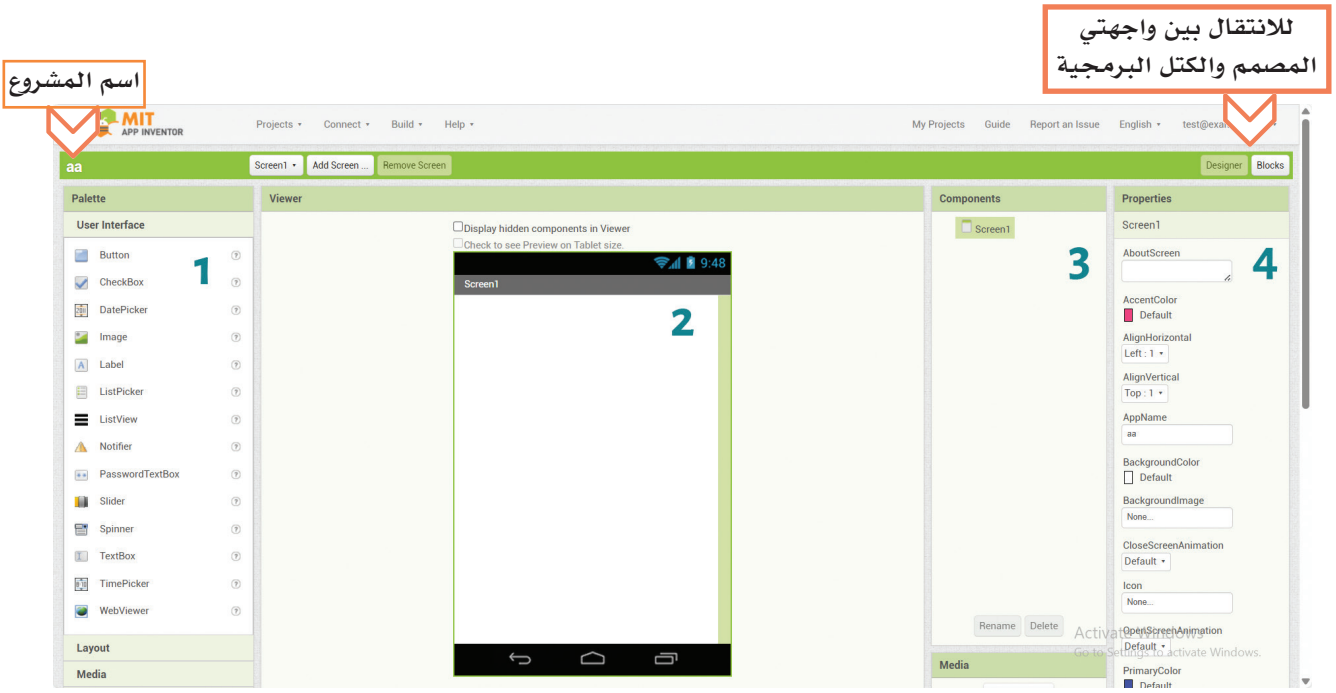
يتم من خلالها بناء واجهة التطبيق عبر سحب و إفلات المكونات المرئية مثل (الأزرار، الخلفيات، الصور، والنصوص) إلى منطقة العمل وضبط خصائصها من لوحة الخصائص مثل (الألوان، العناوين، والمحاذاة).

- **واجهة الكتل البرمجية Blocks:**

يتم من خلالها إضافة المنطق البرمجي و الوظائف إلى التطبيق باستخدام لغة برمجة مرئية والتي يمكن ربطها معًا و ترتيبها في مساحة عمل مرئية لوصف سلوك التطبيق وكيفية استجابته للأحداث.

أولا: واجهة المصمم Designer

- تستخدم لبناء الشكل الخارجي للتطبيق، وتتكون من أربعة أجزاء رئيسية كما في الشكل التالي:



شكل (3): واجهة المصمم في منصة MIT App Inventor.

1. منطقة الأدوات Palette:

الجزء الذي يحتوي على مجموعات المكونات التي يمكن سحبها إلى منطقة العمل ويتغير محتواها حسب السياق التالي:

- **واجهة المصمم Designer:** تُعرض مكونات واجهة المستخدم مثل الأزرار، الصور، مربعات النص، القوائم، إلخ.
- **واجهة الكتل البرمجية Blocks:** تُعرض الكتل البرمجية الخاصة بالمكونات التي تم إضافتها، مثل كتل الأحداث، الإجراءات، القيم، والمنطق.

2. منطقة العمل Viewer:

الجزء المرئي الذي تظهر فيه واجهة المصمم للتطبيق، وما يتم ادراجه من المكونات أو الكتل البرمجية، حيث يتم ترتيب المكونات بصرياً أثناء التصميم، وذلك لمعاينة واجهة التطبيق الخاصة بالهاتف قبل النشر.

3. منطقة المكونات components:

منطقة تظهر فيها جميع المكونات المضافة إلى شاشة التطبيق الحالية ويمكن من خلالها تحديد أي مكون لتعديل خصائصه أو حذفه.

4. منطقة الخصائص Properties:

منطقة تستخدم لتعديل خصائص المكون المحدد الذي تم سحبه إلى منطقة العمل، مثل الألوان والخطوط والحجم، حيث تختلف محتويات هذه المنطقة باختلاف المكون المحدد.



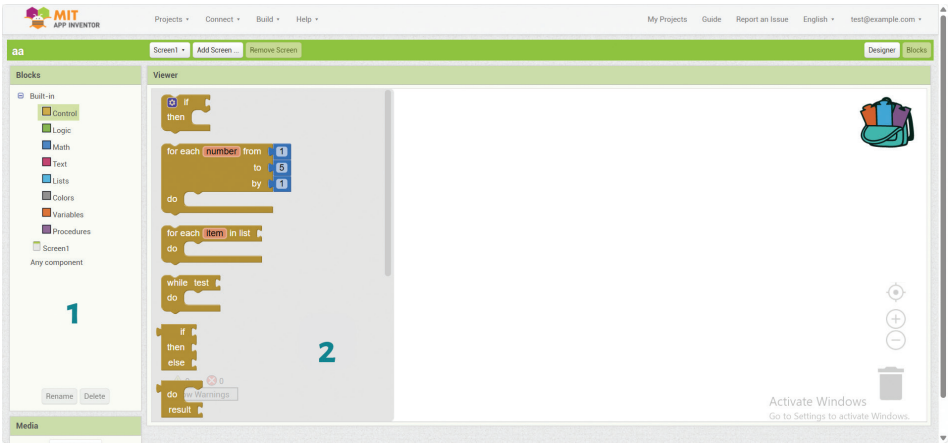
يوجد بجانب كل مكون في منطقة الأدوات علامة استفهام (؟) هي عبارة عن توضيح مبسط عن المكون ووظيفته وطريقة استخدامه.



شكل (4): إظهار تعليمات المكون وطريقة استخدامه

ثانياً: واجهة الكتل البرمجية Blocks

البيئة التي يُبنى فيها منطق التطبيق باستخدام برمجة مرئية تعتمد على السحب والإفلات، دون الحاجة إلى كتابة تعليمات برمجية نصية. وتُستخدم لتحديد سلوك التطبيق، أي كيف يتفاعل مع المستخدم عند حدوث أحداث معينة مثل الضغط على زر أو إدخال نص، وتنقسم هذه الواجهة إلى:



شكل رقم (5): واجهة الكتل البرمجية.

1. منطقة الكتل البرمجية Blocks:

منطقة تحتوي على تصنيفات الكتل البرمجية، مثل: التحكم Control، المنطق Logic، الحساب Math، النصوص Text، القوائم List، المتغيرات Variables، الإجراءات Procedures، بالإضافة إلى كتل خاصة بكل عنصر تمت إضافته في واجهة المصمم.

2. منطقة العمل Viewer:

هي المنطقة التي يتم فيها تصميم واجهة المستخدم User Interface (UI) بطريقة مرئية وتفاعلية، حيث يتم فيها سحب الكتل وترتيبها وربط الأحداث بالإجراءات البرمجية لتحديد سلوك التطبيق، وتحتوي على أدوات التكبير والتصغير لتسهيل ترتيب الكتل وتنظيمها.

تذكر

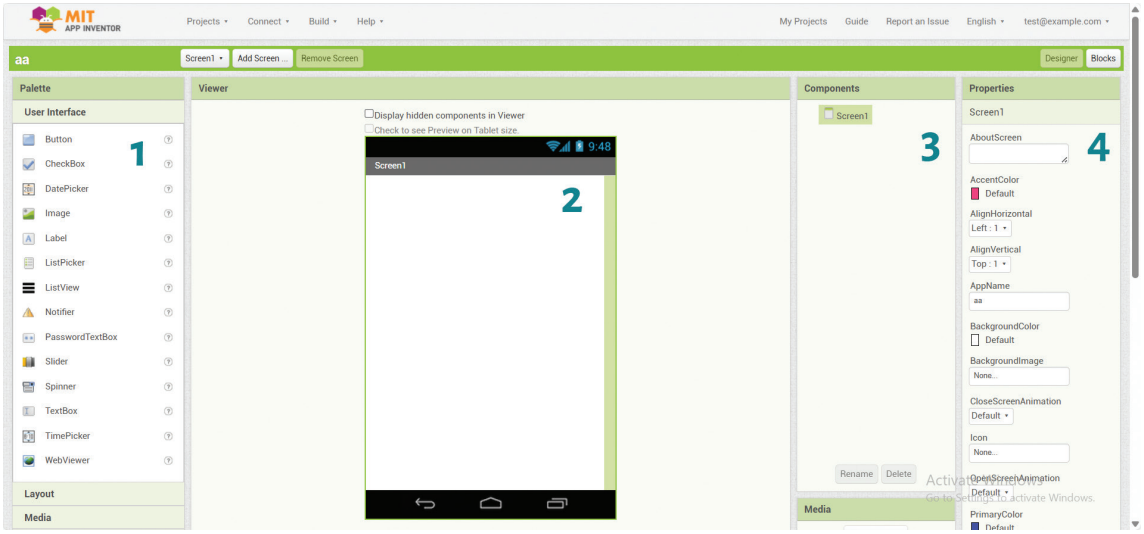
- واجهة المصمم: تُستخدم لترتيب المكونات المرئية وتنسيق مظهر التطبيق.
- واجهة الكتل البرمجية: تُستخدم لبناء منطق التطبيق وسلوكه بالأحداث والإجراءات البرمجية.



التطبيـــق

ورقة عمل (1)

من خلال الشكل التالي اكتب الرقم المناسب أمام كل عبارة مما يلي:



المنطقة	الرقم المناسب
منطقة الأدوات Palette	
منطقة الخصائص Properties	
منطقة العمل Viewer	
منطقة المكونات components	



عبر عن رأيــك



أتعرف على مفهوم تطوير تطبيقات الأجهزة الذكية.	☐	☐ ☐
أتعرف على دورة حياة تطوير البرمجيات.	☐	☐ ☐
أميز بين أنظمة التشغيل المختلفة للأجهزة الذكية.	☐	☐ ☐
أصف المكونات الرئيسة لواجهة المصمم Designer.	☐	☐ ☐
أصف المكونات الرئيسة لواجهة الكتل البرمجية Blocks.	☐	☐ ☐
أميز بين بيئة البرمجة النصية وبيئة البرمجة المرئية.	☐	☐ ☐



ملاحظات المعلم

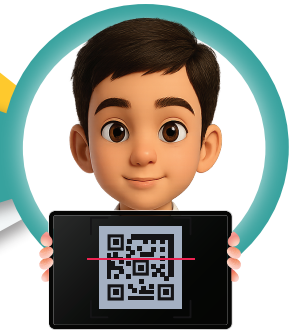
اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

التعامل مع برنامج / منصة MIT App Inventor

نواتج التعلم

- التعرف على برنامج / منصة MIT App Inventor.
- التعامل مع برنامج / منصة MIT App Inventor.
- تمييز الفرق بين العمل من خلال المنصة Platform والنسخة المحمولة Portable.
- إنشاء مشروع جديد من خلال برنامج / منصة MIT App Inventor.
- التعامل مع مكونات التصميم في منطقة الأدوات Palette.
- تصميم واجهة تطبيق بسيطة باستخدام مكونات واجهة المستخدم.
- إدراج المكونات في منطقة العمل Viewer بالسحب والإفلات .
- تعديل خصائص مكونات التصميم (مثل اللون، الحجم، النص).
- استخدام التخطيط Layout لتنظيم المكونات داخل واجهة المستخدم بطريقة مرتبة وواضحة.
- إظهار الاهتمام بتعلم تصميم التطبيقات البرمجية.
- الالتزام بالدقة والتنظيم أثناء العمل على تصميم واجهة تطبيق.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــــم

طرق التعامل مع برنامج / منصة MIT App Inventor

الطريقة الأولى: المنصة Platform:

الدخول إلى الموقع الرسمي من خلال الرابط:



http://ai2.appinventor.mit.edu

تظهر الواجهة الرئيسية والتي من خلالها يمكن:

- إنشاء مشروع جديد Start new project.
- فتح مشروع سابق My projects.
- عرض التطبيقات النموذجية Gallery.

لاحظ

في حالة استخدام المنصة Platform يتطلب استخدام حساب Google لتسجيل الدخول وحفظ المشاريع بشكل آمن وسحابي، مما يُتيح الوصول إليها من أي جهاز وفي أي وقت.

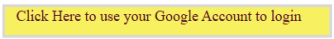
الطريقة الثانية: النسخة المحمولة Portable:

- الدخول باستخدام البريد الإلكتروني الافتراضي الموجود في واجهة التطبيق:



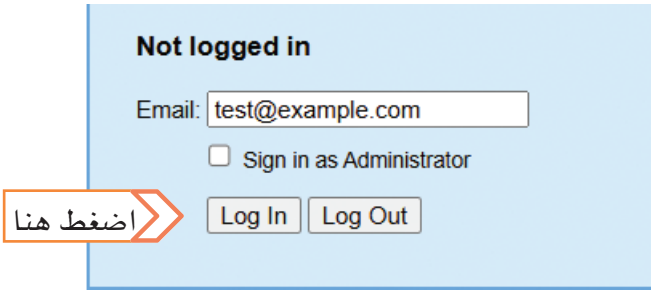
شكل رقم (6): شاشة الدخول للمنصة Portable.

بعد الضغط على الرابط الموضح بالشكل السابق





- يتم الانتقال للصفحة التالية:



شكل رقم (7): شاشة الدخول للمنصة Portable

ثم الضغط على زر Log In للدخول إلى الواجهة الرئيسية للمنصة.



نشـــاط 1

المقارنة بين منصة MIT App Inventor Platform ونسخة MIT App Inventor Portable.

الحاجة للاتصال بالإنترنت

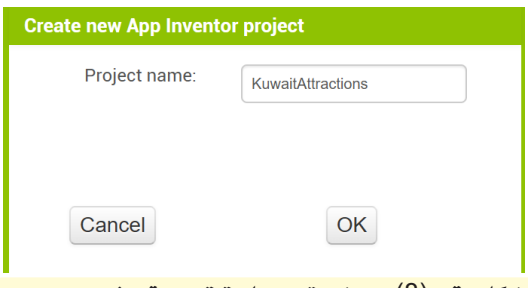
MIT App Inventor Platform

MIT App Inventor Portable

إنشاء مشروع جديد.

يمكـــن إنشـــاء تطبيـــق جديـــد مـــن خلال برنامـــج / منصـــة MIT App Inventor وذلـــك باتبـــاع الخطـــوات التاليـــة:

- الدخول للبرنامج / للمنصة.
- اختيار Start New project من قائمة Projects.
- تظهر شاشة فرعية Create new App Inventor project:
  - كتابة اسم المشروع Project name، وليكن KuwaitAttractions.
  - الضغط على زر Ok.

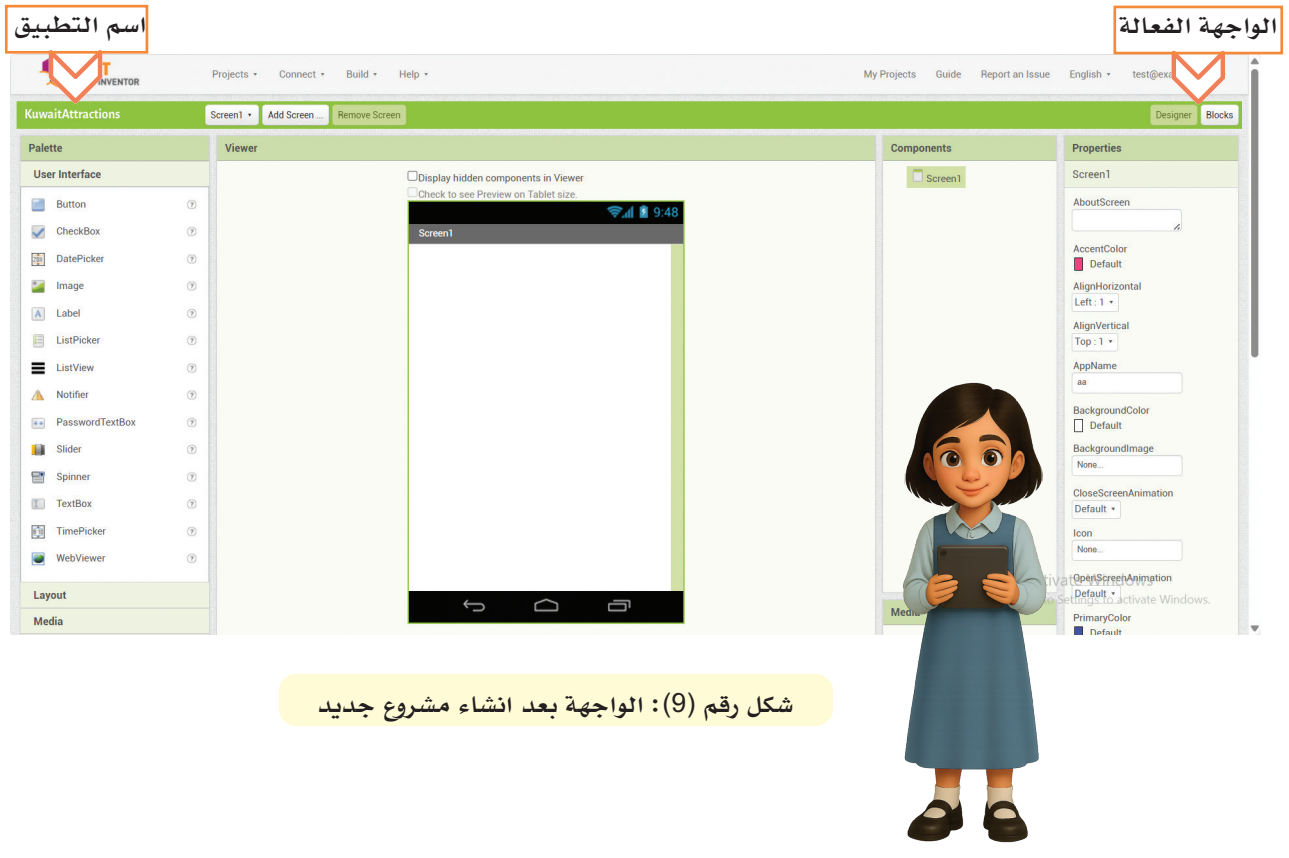


شكل رقم (8): صندوق محاورة تسمية مشروع جديد





■ تظهر واجهة البرنامج / المنصة كما يلي:



شكل رقم (9): الواجهة بعد انشاء مشروع جديد

التعامل مع مكونات التصميم في منطقة الأدوات palette

تحتوي معظم التطبيقات على مجموعة من المكونات مثل (الصور - الازرار - النصوص .....) في برنامج / منصة MIT App Inventor مجموعة متنوعة من مكونات التصميم لبناء التطبيق، وعند تصميم واجهة التطبيق يجب أن تتناسب مع احتياجات المستخدم وطبيعة التطبيق.

وفيما يلي بعض تصنيفات المكونات في المنصة:

مكونات واجهة المستخدم
User Interface components

مثل الأزرار، مربعات النص، الصور.

مكونات التخطيط
Layout components

مثل الترتيب الأفقي والرأسي.

مكونات الوسائط
Media components

مثل الصوت، الفيديو، الكاميرا.

مكونات الرسم والحركة
Drawing and Animation components

مثل اللوحة القماشية Canvas والكائنات المتحركة Sprite.

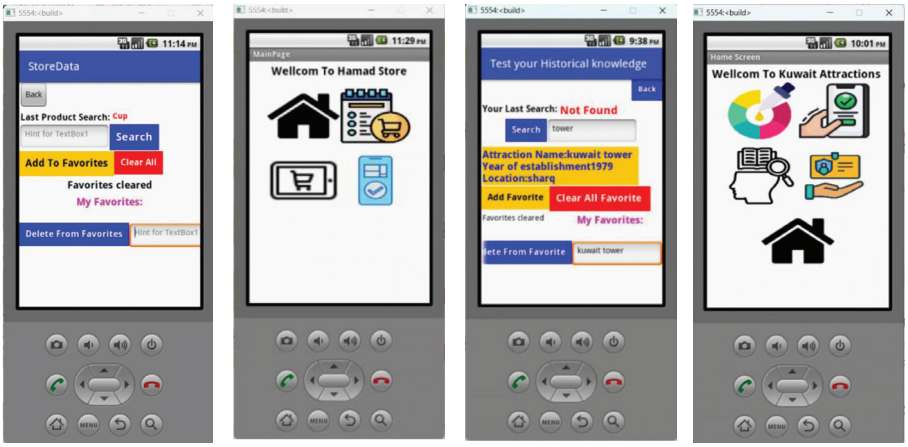
يمكن استكشاف العديد من المكونات الأخرى من خلال البرنامج / المنصة ، وسوف نستعرض العديد منها لاحقاً.

بعض مكونات واجهة المستخدم User Interface components ووظائفها.

زر	صورة	تسمية	مربع نص	قوائم
تنفيذ الكتلة البرمجية عند الضغط عليه.	عرض الصور والرسوم المتحركة الأساسية.	إظهار النصوص الثابتة مثل العناوين أو التعليمات.	إدخال النصوص.	انشاء قوائم بسيطة من السلاسل النصية.
Button	Image	Label	TextBox	ListView

تطبيق KuwaitAttractions

يهدف هذا التطبيق إلى تسليط الضوء على المعالم السياحية في دولة الكويت بطريقة مبسطة وتفاعلية، تُمكّن المستخدم من التعرف على هذه المعالم بسهولة، إلى جانب توفير معلومات إثرائية تزيد من معرفته الثقافية والجغرافية حول دولة الكويت.
و سيتم بناء التطبيق من خلال تنفيذ مجموعة من المهام في كل نشاط من الأنشطة التالية:



شكل رقم (10): بعض شاشات تطبيق KuwaitAttractions بعد استكماله



نشاط 2

مهمة النشاط:

تصميم واجهة ترحيبية لتطبيق KuwaitAttractions تُظهر شعار دولة الكويت، عنوان نصي، وزر للانتقال ، مع تعديل خصائص كل مكوّن بما يتناسب مع التصميم.

تنفيذ النشاط :

- تشغيل برنامج / منصة MIT App Inventor.

1. إنشاء تطبيق جديد باسم KuwaitAttractions، وذلك من خلال تنفيذ الخطوات التالية:

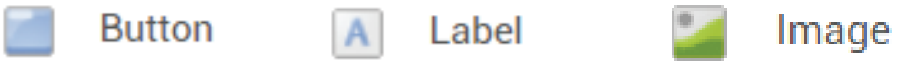
- من قائمة Projects يتم اختيار Start new project.

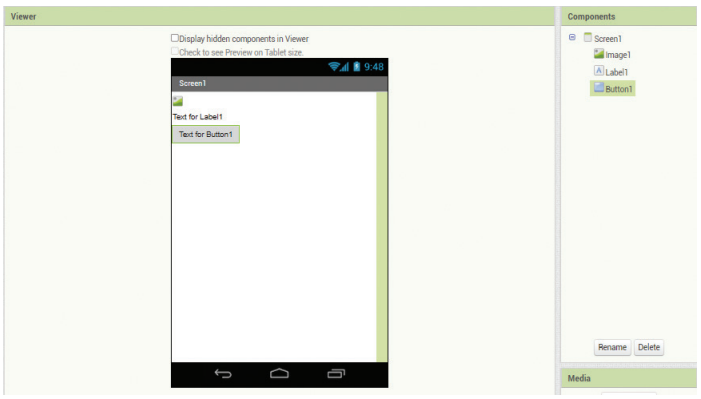
في صندوق المحاورة يتم :

- كتابة اسم المشروع في خانة Project name : KuwaitAttractions
- الضغط على زر Ok

تفعيل واجهة المصمم Designer وتنفيذ الخطوات التالية:

2. من لوحة الأدوات Palette سحب المكونات التالية إلى منطقة العمل Viewer من تصنيف User Interface:



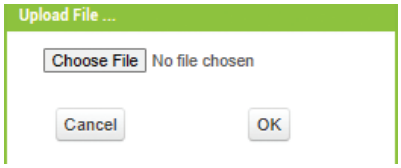


شكل رقم (11): منطقة العمل بعد إدراج المكونات

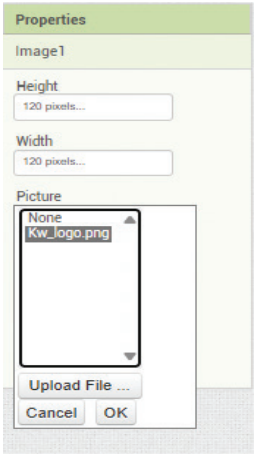
3. تحديد المكون المطلوب، ثم تعديل الخصائص المطلوبة من منطقة Properties كالتالي:

- تحديد مكون الصورة Image1:
  - تعديل أبعاد الصورة من الارتفاع Hight : 120 pixels.
  - تعديل أبعاد الصورة من العرض Width : 120 pixels.
  - من خاصية الصورة Picture: يتم الضغط على Upload File يظهر صندوق المحاورة:

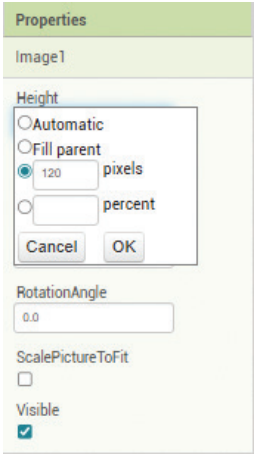
- من Choose File يتم اختيار صورة الشعار Kw_logo من مجلد Images ثم الضغط على زر OK.



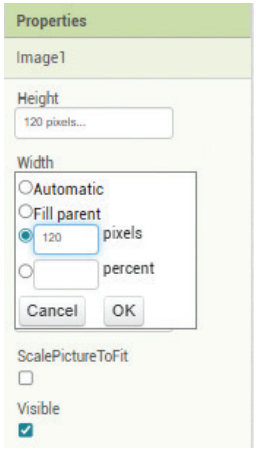
شكل رقم (12): صندوق محاورة Upload File



إدراج صورة للمكون Image1



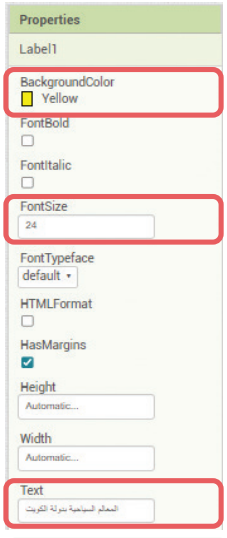
تغيير الارتفاع للمكون Image1



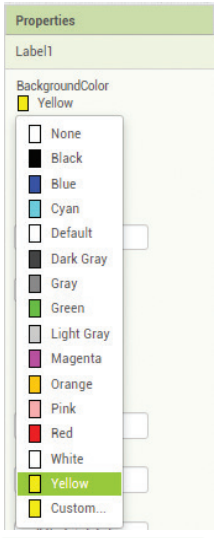
تغيير العرض للمكون Image1

شكل رقم (13): إضافة الصورة للمكون Image1 وتعديل أبعادها

- تحديد التسمية Label1 ثم من منطقة الخصائص يتم:



تغيير حجم الخط و النص للمكون Label1



تغيير لون خلفية للمكون Label1

تغيير لون الخلفية
BackgroundColor: الأصفر Yellow.

تغيير حجم الخط FontSize : 24.

كتابة النص Text:
المعالم السياحية بدولة الكويت.

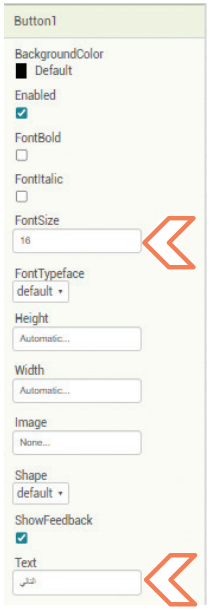


شكل رقم (14): تغيير خصائص Label1

- تحديد الزر Button1 ثم من الخصائص يتم:



شكل رقم (15): تغيير خصائص Button1



تغيير حجم الخط، والنص للمكون Button1

تغيير حجم الخط FontSize : 16.

كتابة النص Text : التالي.

4. حفظ التطبيق باسم KuwaitAttractions1 من قائمة Projects.



شكل رقم (16) : معاينة شاشة التطبيق

أساسيات الترتيب الأفقي والرأسي HorizontalArrangement & VerticalArrangement

الترتيب الأفقي يشبه ترتيب الكتب على الطاولة أفقياً، بحيث تُوضع الكتب جنبًا إلى جنب من اليسار إلى اليمين أو العكس، مما يجعل الكتب تظهر صفًا واحدًا يمتد أفقيًا دون تداخل رأسي.

أما الترتيب الرأسي فيشبه تراكم الصناديق فوق بعضها البعض في شكل عمود، حيث تُرتب العناصر من الأسفل إلى الأعلى، مما يشكل عمودًا يمتد رأسيًا ويسمح بالتمرير إذا زاد العدد.

استخدام الترتيب الأفقي والترتيب الرأسي في التصميم:

عند تصميم التطبيق فإن الترتيب الأفقي يساعد في التوزيع الجانبي لتجنب الازدحام، بينما يركز الترتيب الرأسي على التدفق الطولي لتسهيل القراءة في الشاشات المحدودة.

- الترتيب الأفقي HorizontalArrangement:

من مكونات التخطيط Layout في البرنامج / المنصة ، ويُستخدم لترتيب و تنظيم المكونات داخل التطبِّيق في صف واحد يمتد أفقيًا من اليسار إلى اليمين أو العكس حسب إعدادات اللغة أو التصميم ، مما يجعله مناسباً في تصميم القوائم الأفقية أو شريط أدوات في واجهات التطبيقات.

- الترتيب الرأسي VerticalArrangement:

من مكونات التخطيط Layout في البرنامج / منصة، ويُستخدم لترتيب وتنظيم المكونات داخل التطبيق في عمود يمتد رأسيًا من الأعلى إلى الأسفل أو العكس، مما يجعله مناسباً لعرض خطوات، تعليمات، أو نماذج إدخال بيانات.



نشاط 3

مهمة النشاط:

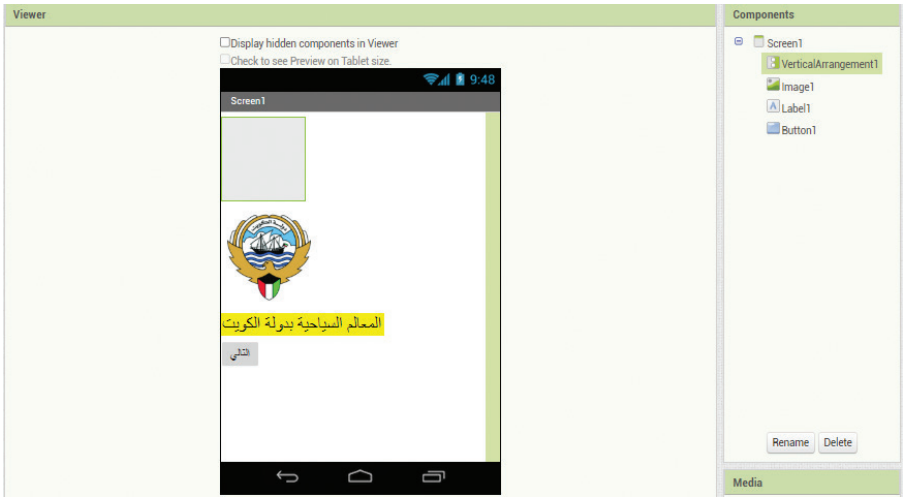
استكمال تطبيق KuwaitAttractions1 وذلك بإعادة ترتيب مكونات الشاشات من خلال الترتيب الرأسي VerticalArrangement.

تنفيذ النشاط:

- تشغيل منصة MIT App Inventor.
- استدعاء التطبيق KuwaitAttractions1، ثم تفعيل واجهة المصمم Designer وتنفيذ الخطوات التالية:

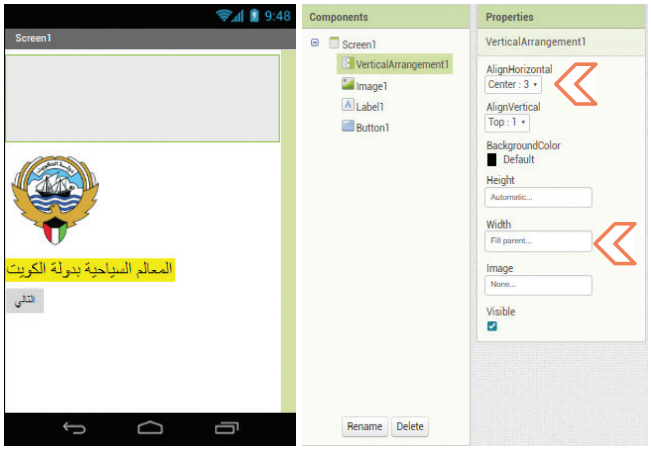
1. من لوحة الأدوات ثم من تصنيف Layout :

سحب المكون: الترتيب الرأسي VerticalArrangement إلى منطقة العمل Viewer.



شكل رقم (17) : شاشة التطبيق بعد إضافة المكون VerticalArrangement

2. تحديد المكون VerticalArrangement1 ثم تغيير خصائصه في منطقة Properties كالتالي:



شكل رقم (18) : تعديل خصائص VerticalArrangement1



- تعديل خصائص المكون VerticalArrangement1

3. إضافة جميع مكونات التطبيق التي تم إدراجها سابقا داخل VerticalArrangement1 بالسحب والإفلات.

4. حفظ التطبيق باسم KuwaitAttractions2 من القائمة Projects ثم اختيار Save project as.



شكل رقم (19): معاينة واجهة التطبيق بعد التنفيذ.

لاحظ

تعديل الخصائص فيما يتعلق بالترتيب الرأسي والأفقي يظهر في صورة أرقام حسب الجدول التالي:

الترتيب الأفقي AlignHorizontal

رقم	
1	المحاذاة Right
2	المحاذاة Left
3	المحاذاة Center

الترتيب الرأسي AlignVertical

رقم	
1	المحاذاة Top
2	المحاذاة Center
3	المحاذاة Bottom





التطبيـــق

ورقة عمل (2)

من خلال دراستك لبرنامج / منصة MIT App Inventor نفذ الخطوات التالية:

1. أنشئ مشروعًا جديدًا في MIT App Inventor باسم: HamadStore.
2. في واجهة المصمم أضف المكونات: VerticalArrangement - Image - Label - Button في شاشة التطبيق في منطقة العمل، ثم نفذ التالي:

- عدّل خصائص المكون الصورة: Image1
  - الارتفاع Height : 200 pixels
  - العرض Width : 200 pixels
  - صورة شعار التطبيق Hamadlogo من مجلد images.



- عدّل خصائص المكون التسمية Label1:
  - سُمك الخط FontBold :
  - حجم الخط FontSize : 30
  - النص Text : Hamad Store

- عدّل خصائص المكون الزر : Button1
  - لون الخلفية BackgroundColor: أزرق Blue
  - سُمك الخط FontBold :
  - حجم الخط FontSize : 24
  - النص Text : Shop Now
  - لون النص TextColor : أبيض White



- عدّل خصائص المكون الترتيب الرأسي : VerticalArrangement1.
  - المحاذاة الأفقية AlignHorizontal: center:3.
  - العرض Width : Fill parent.

3. احفظ التطبيق باسم HamadStore2.



في وقت فراغك

استعرض واجهة المصمم Designer في MIT App Inventor، ثم استكشف ثلاثة مكونات، و دوّن اسم كل مكون ووظيفته الأساسية داخل التطبيق.

المكون	وظيفته



عبر عن رأيــك

أتعرف على برنامج / منصة MIT App Inventor.	☐	☐	☐
أميز بين العمل من خلال منصة Platform والنسخة المحمولة Portable.	☐	☐	☐
أنشئ مشروعاً جديداً من خلال برنامج / منصة MIT App Inventor.	☐	☐	☐
أتعامل مع مكونات التصميم في منطقة الأدوات palette.	☐	☐	☐
أصمم واجهة تطبيق بسيطة باستخدام مكونات واجهة المستخدم.	☐	☐	☐
أدرج المكونات في منطقة العمل Viewer بالسحب والإفلات.	☐	☐	☐
أعدل خصائص مكونات التصميم (مثل اللون، الحجم، النص).	☐	☐	☐
أستخدم التخطيط Layout لتنظيم المكونات داخل واجهة المستخدم بطريقة مرتبة وواضحة.	☐	☐	☐



ملاحظات المعلم

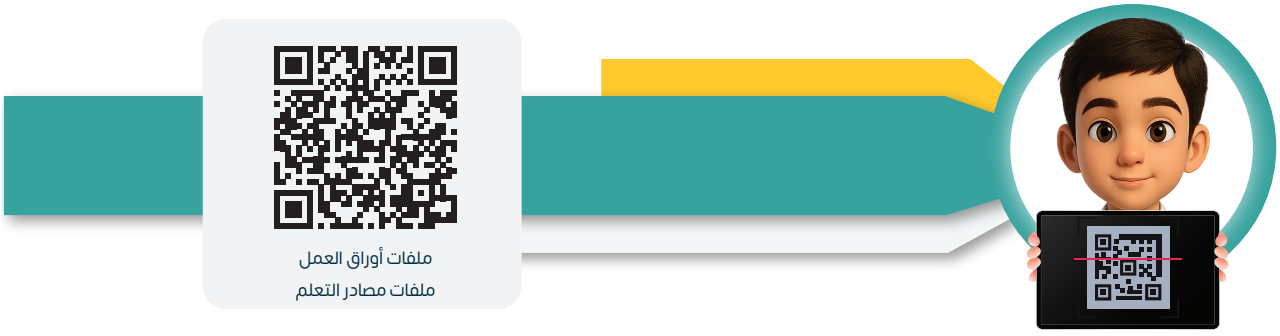
اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

واجهة الكتل والأحداث Blocks & Events Editor

نواتج التعلم

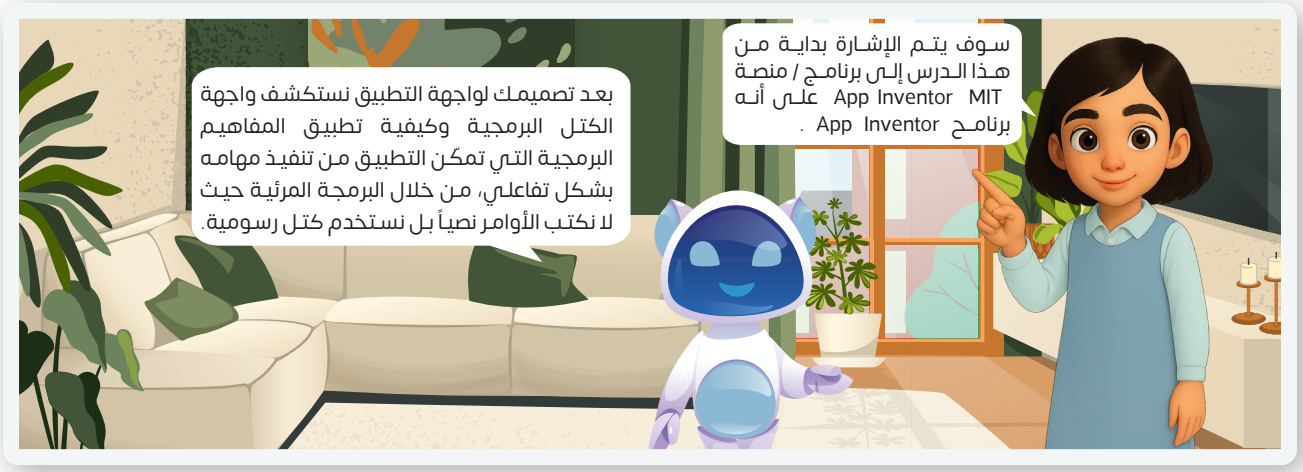
- التعرف على مفهوم البرمجة بالكتل.
- التعرف على أنواع الكتل الأساسية.
- التمييز بين مفهومي كتل الأحداث وكتل الإجراءات.
- التعرف على مفهوم الحدث البرمجي ودوره في استجابة التطبيق لتفاعل المستخدم.
- تمييز الأنواع المختلفة من كتل الأحداث واستخداماتها المختلفة حسب التفاعل المطلوب.
- تفسير الإجراء البرمجي كمجموعة من التعليمات القابلة لإعادة الاستخدام داخل البرنامج.
- بناء إجراءات برمجية بسيطة وتنظيم التعليمات البرمجية بشكل منطقي.
- ربط المكونات المختلفة بكتل برمجية مناسبة لتحقيق وظائف محددة في التطبيق.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــــم

أنواع الكتل البرمجية الأساسية:

1 كتل الأحداث Event Handlers

كتل برمجية أساسية تحدد استجابة التطبيق للأحداث الخارجية أو الداخلية، مثل الضغط على زر معين فيتم الاستجابة لهذا الحدث بدلاً من التنفيذ الخطي التقليدي.

أنواع الأحداث الرئيسية :

تنقسم الأحداث في برنامج App Inventor إلى نوعين رئيسيين هما :

الأحداث الناتجة عن المستخدم
user-initiated events

مثل الضغط على زر

الأحداث التلقائية
automatic events

مثل تشغيل مؤقت
أو تلقي رسالة نصية

مفهوم الأحداث :

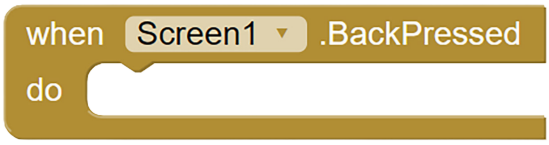
تنفيذ أوامر محدده عند إجراء حدث معين.

أمثلة على كتل الأحداث:

- عند الضغط على الزر When Button Clicked



- عند الضغط على الشاشة When Screen BackPressed



2 كتل الإجراءات Procedures

تسلسلات برمجية (مجموعة مترابطة من الأوامر البرمجية) مسماة تُستخدم لتنفيذ مهمة محددة داخل التطبيق، حيث يمكن استدعاؤها في أي وقت وفي أكثر من مكان داخل التطبيق، مما يسمح بإعادة استخدامها وتنظيم البرنامج بشكل أفضل بدلاً من تكرار التعليمات البرمجية.

■ أنواع الإجراءات الرئيسية :

تنقسم الإجراءات بناءً على قدرتها في التعامل مع المدخلات إلى نوعين رئيسيين:

■ إجراء بسيط Simple Procedure:

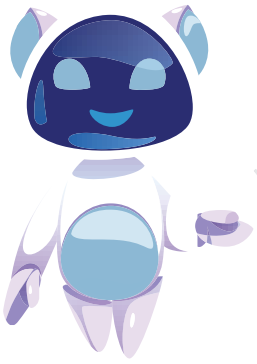
ينفذ مهمة محددة دون الحاجة إلى بيانات إضافية، وتُعرّف هذه التصنيفات باستخدام كتل برتقالية اللون دون وجود أماكن مخصصة لاستقبال البيانات.

■ إجراء مع مدخلات Procedure with Inputs:

كتلة برمجية تحتوي على أماكن مخصصة لاستقبال بيانات تُعرف بالمعاملات Arguments، تُستخدم هذه البيانات أثناء تنفيذ الإجراء، مما يسمح بتمرير معلومات ديناميكية تختلف حسب السياق، ويُسهم ذلك في أن جعل التعليمات البرمجية أكثر مرونة لإعادة الاستخدام.

لاحظ

إجراء عام يقبل بيانات إدخال ديناميكية، مما يجعله قابلاً لإعادة الاستخدام في سياقات مختلفة دون الحاجة إلى تعديل التعليمات البرمجية الأساسية.



للتعامل مع الكتل البرمجية يجب تفعيل واجهة الكتل البرمجية من خلال الضغط على زر Blocks الموجود على واجهة البرنامج .



نشـــاط 1

مهمة النشاط :

كتابة المسمى الدال على كل مفهوم مما يلي:

المفهوم	مجموعة أوامر برمجية يتم تنفيذها معا.	تنفيذ أوامر محدده عند إجراء حدث معين.
اسم المفهوم		

3 التعامل مع مكونات الكتل البرمجية

تحتوي أغلب التطبيقات على مجموعة من المكونات مثل (الصور - الازرار - النصوص .....) يتم ربطها مع مجموعة من الكتل البرمجية حتى يلبي التطبيق احتياجات المستخدم المختلفة بطريقة مرنة وسهلة.

■ أنواع الكتل البرمجية ووظائفها :

يوفر برنامج App Inventor مجموعة متنوعة من تصنيفات الكتل البرمجية ، وفيما يلي بعض أنواعها :

■ كتل التحكم Control :

توفر أدوات للتحكم في تدفق البرنامج.

كتل المنطق Logic :

تشمل قيمًا منطقية أساسية مثل true و false، بالإضافة إلى عمليات منطقية مثل NOT، OR، AND، ومقارنات مثل = لربط الشروط المعقدة.

كتل الحساب Math:

يتيح إجراء عمليات رياضية أساسية مثل الجمع، الطرح، الضرب، والقسمة، بالإضافة إلى وظائف متقدمة مثل الجذر التربيعي أو توليد أرقام عشوائية.

كتل النصوص Text:

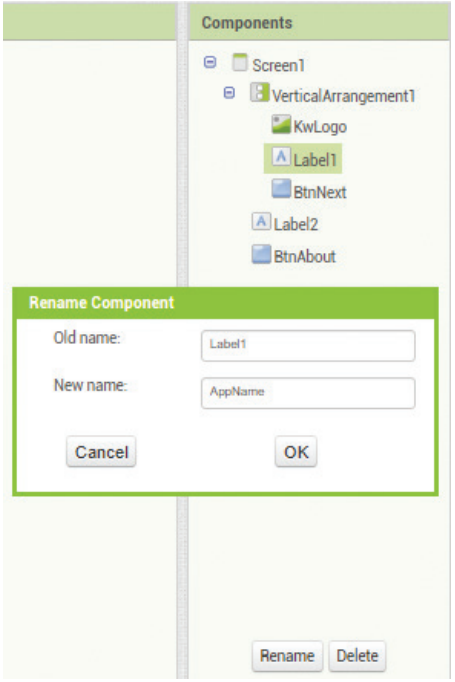
تركز على إنشاء ومعالجة النصوص، مثل ربط أجزاء نصية (join)، فصلها (split)، أو مقارنتها لعرضها في مكونات الواجهة.

يمكن استكشاف العديد من الكتل البرمجية الأخرى من خلال برنامج App Inventor MIT، وسوف نستعرض لاحقاً العديد منها.

تغيير أسماء المكونات داخل التطبيق:

خطوات تغيير أسماء المكونات داخل التطبيق وذلك حتى يسهل التعامل معها في منطقة الكتل البرمجية:

1	تحديد المكون في منطقة المكونات Components.
2	اختيار الزر Rename .
3	كتابة الاسم الجديد في خانة New name.
4	الضغط على زر Ok.



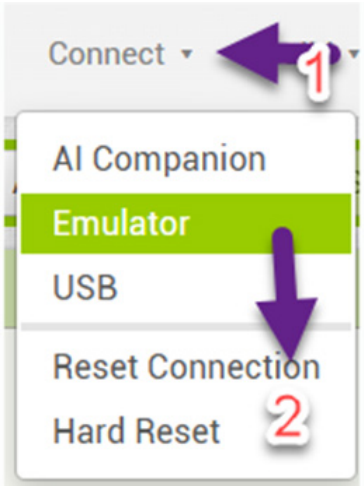
شكل رقم (20): تغيير اسم المكون

المحاكي في برنامج App Inventor

يوفر البرنامج إمكانية عرض التطبيق مباشرة على المحاكي أثناء عملية الإنشاء مما يساعد على اختبار الكتل البرمجية والتأكد من عملها بشكل صحيح قبل نشر التطبيق.

تشغيل المحاكي :

- من قائمة Connect يتم اختيار Emulator.



شكل رقم (21): تشغيل المحاكي



نشاط 2

مهمة النشاط:

تطوير تطبيق KuwaitAttractions2. بحيث يتم إدراج المكونات Label - Button لإظهار معلومات عن التطبيق في ضوء الخطوات التي سيتم عرضها لتنفيذ النشاط.

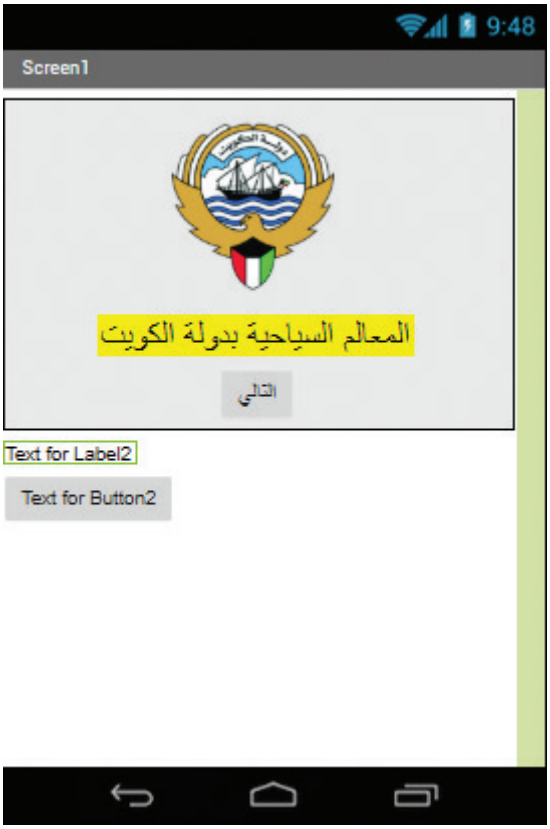
تنفيذ النشاط :

وذلك باتباع الخطوات التالية :

تشغيل برنامج App Inventor ، **ثم استدعاء التطبيق** KuwaitAttractions2 **من مجلد** Activity.

■ أولاً : تحديد واجهة المصمم Designer :

1 . من منطقة الأدوات ثم تصنيف User Interface **يتم إدراج مكونات جديدة :**

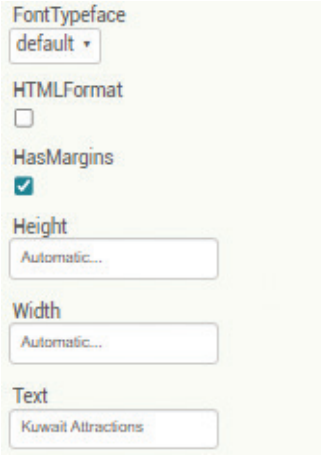




شكل رقم (22) : الشاشة بعد إضافة المكونات

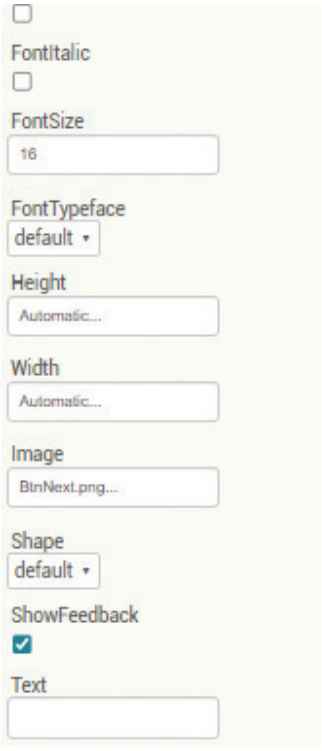
2. تحديد المكون المطلوب تغيير إعداداته في منطقة المكونات Components، ثم الانتقال لمنطقة الخصائص Properties :

المكون	الخاصية	القيمة
Label1	النص Text	تعديل الاسم ليصبح: Kuwait Attractions

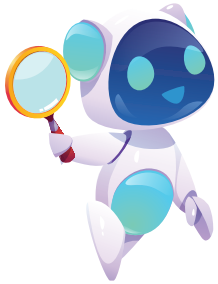


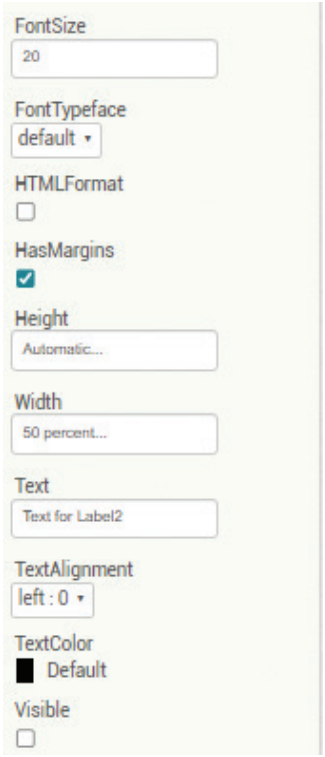
شكل رقم (23): تعديل خصائص مكون Lablel1

المكون	الخاصية	القيمة
Button1	الصورة Image	تحميل صورة BtnNext من مجلد images.
	النص Text	حذف كلمة (التالي).



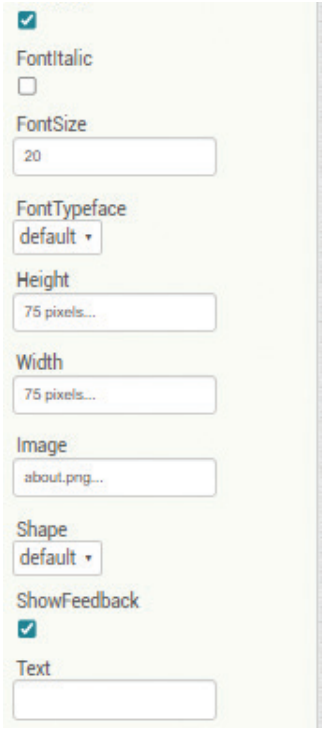
شكل رقم (24): تعديل خصائص مكون Button1





المكون	الخاصية	القيمة
Label2	FontBold النص سميك	☑
	FontSize حجم الخط	20
	Width العرض	Percent 50
	Visible الظهور	☐

شكل رقم (25): تعديل خصائص مكون Label2



المكون	الخاصية	القيمة
Button2	Hight تغيير الارتفاع	75 pixels
	Width العرض	75 pixels
	Image الصورة	about تحميل صورة من مجلد Media

شكل رقم (26): تعديل خصائص مكون Button2

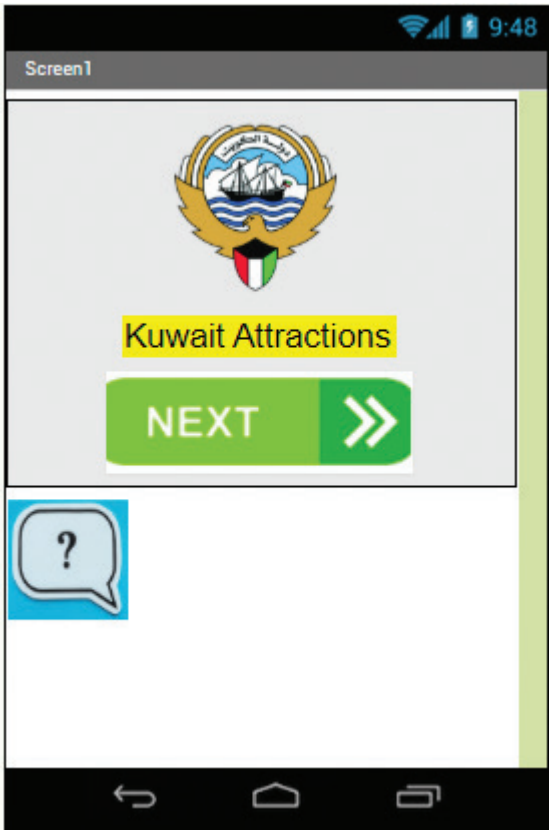


بعد إضافة صورة للزر Button2 يجب التأكد من إزالة النص من خاصية Text لمكون الزر حتى لا يظهر النص فوق الصورة.





شكل رقم (27): حذف النص من مكون Button2

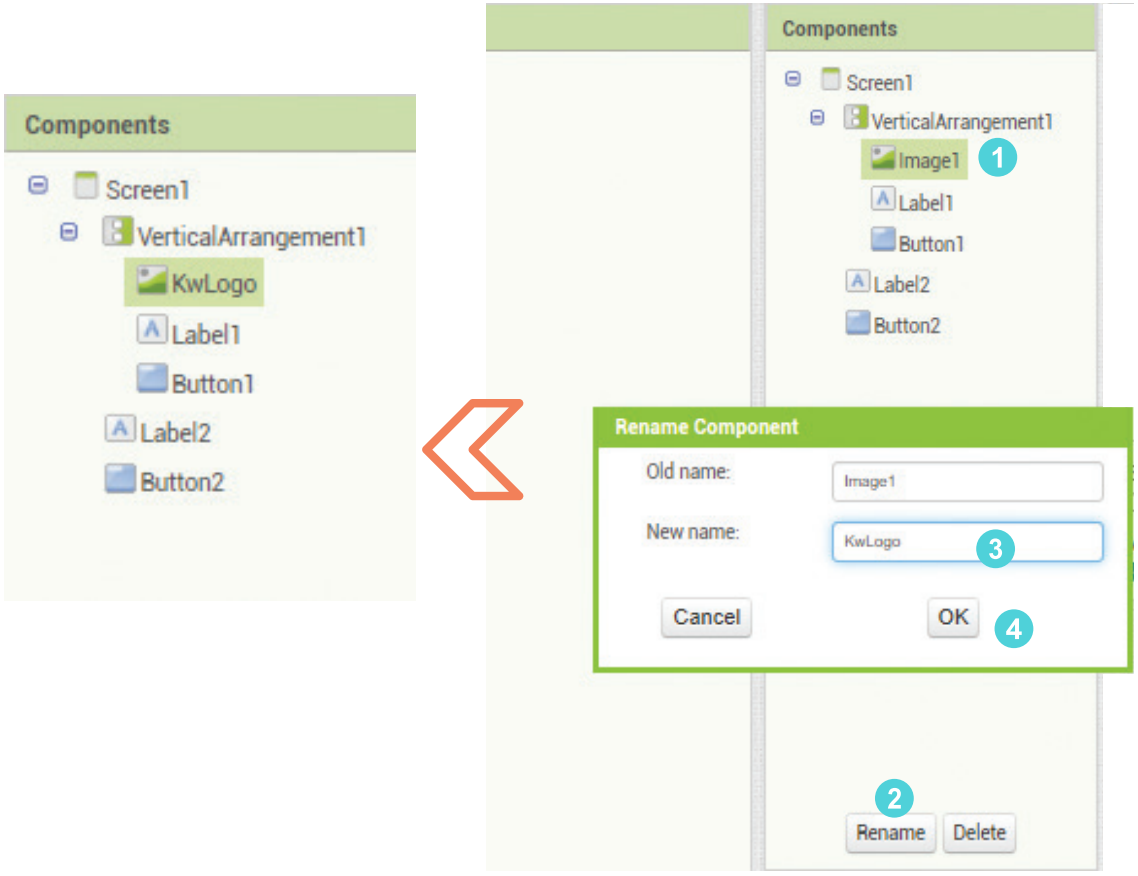


شكل رقم (28): تعديل واجهة التطبيق بعد تغيير خصائص المكونات

تغيير مسميات المكونات التالية:

المكون	Image1	Button1	Button2	Label1	Label2
Rename	KwLogo	BtnNext	BtnAbout	AppName	VData

حيث يتم ذلك باتباع الخطوات التالية :



شكل رقم (29) : خطوات تغيير اسم المكون

1	تحديد المكون المطلوب تغيير اسـمه ثم في منطقة Components.
2	الضغط على زر Rename.
3	من خلال صندوق المحاورة Rename Component يتم كتابة الاسـم الجديد للمكـون المحدد في خانة New name.
4	الضغط على زر Ok.

ثانيا : واجهة الكتل البرمجية Blocks :

4.إضافة الكتل البرمجية لإظهار معلومات التطبيق عند الضغط على الزر وذلك باتباع التالي:

1) لانتقال لواجهة الكتل البرمجية ومن ثم تحديد المكون BtnAbout ، حيث تظهر الكتل البرمجية الخاصة بالمكون



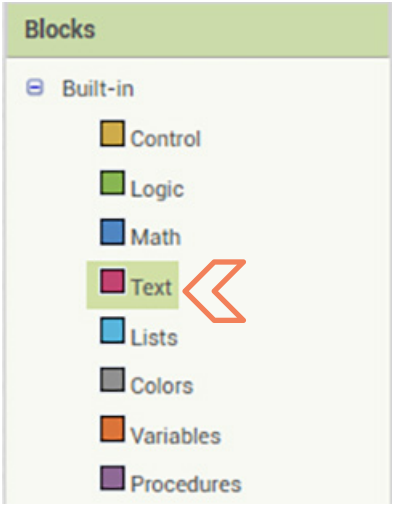
شكل رقم (30) : الكتل البرمجية الخاصة بالمكون Btnabout

2) اختيار الكتلة البرمجية When Btnabout.click ووضعها في منطقة العمل Viewer بالسحب والإفلات



- **لتغيير النص يتم اتباع الخطوات التالية :**
  - تحديد المكون VData في منطقة Blocks.
  - اختيار الكتلة البرمجية





  - تغيير النص داخل المكون VData، من خلال الضغط على Text في منطقة Blocks.
  - إضافة الكتلة البرمجية A Text String



  - الضغط على الكتلة السابقة بالزر الأيسر للفأرة لتفعيل الكتابة داخلها، ومن ثم كتابة رقم إصدار التطبيق «App version 1.1»، ومن ثم يتم دمج الكتلتين معًا بوضع كتلة النص داخل خانة to.

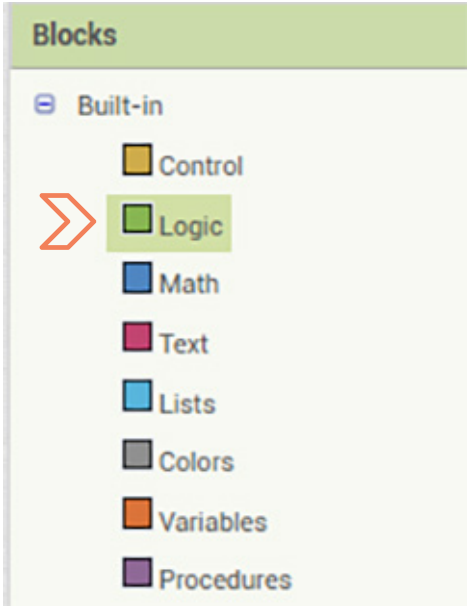


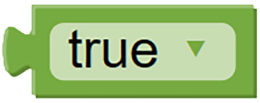
- **لإظهار المكون VData يتم اتباع الخطوات التالية :**
  - اختيار الكتلة البرمجية Set VData.Visible to.



واجهة الكتل والأحداث Blocks & Events Editor

- الضغط على Logic في منطقة Blocks ومن ثم سحب الكتلة البرمجية True إلى منطقة العمل.





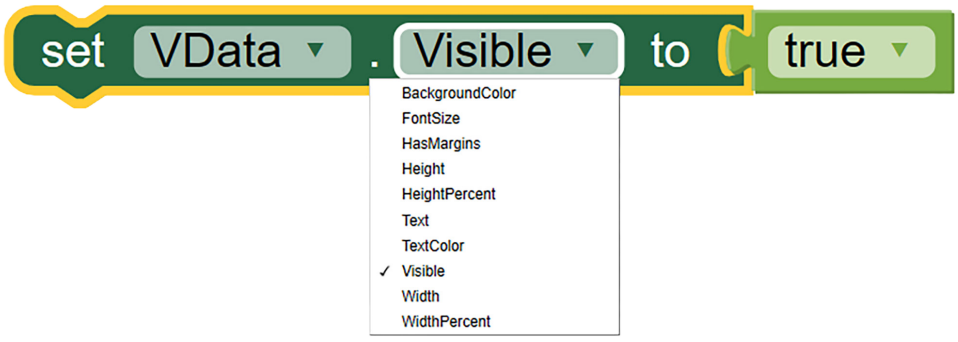
ناقش مع معلمك
لماذا تم اختيار الكتلة البرمجية؟
True

- دمج الكتلتين معًا بوضع كتلة true داخل خانة to .

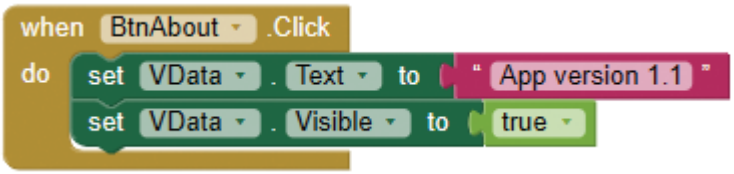


لاحظ

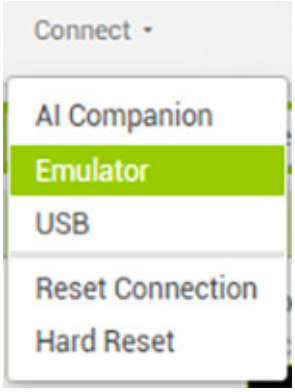
بعض الكتل البرمجية يمكن تعديل وظائفها أو تغيير المكون الخاص بها لسهولة العمل داخل منطقة العرض Viewer من خلال القائمة المنسدلة بجانب الخاصية أو المكون، على سبيل المثال الكتلة البرمجية Set يمكن تغيير إما (Text,Hight,FontSize, ...).



- إضافة الكتل البرمجية بعد تجميعها داخل الكتلة البرمجية للزر Btnabout لتظهر كما يلي :



5. حفظ التطبيق باسم KuwaitAttractions3 ومن ثم عرض التطبيق في شاشة المحاكي Emulator باتباع الخطوات التالية:



- من القائمة Connect يتم اختيار Emulator.

- يبدأ المحاكي التحميل وقد يستغرق دقيقة أو أكثر حسب سرعة الجهاز.



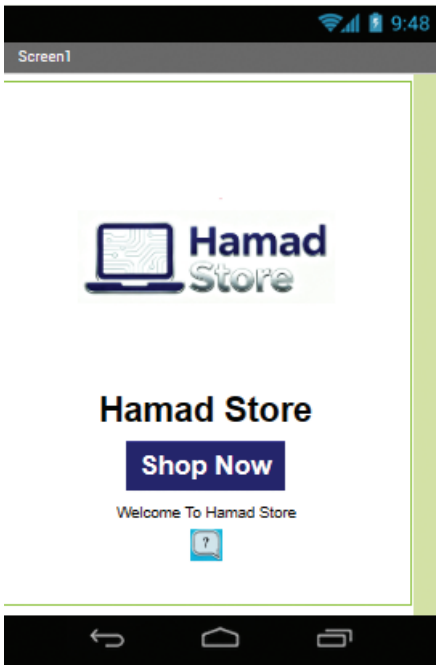
التطبيـــق

ورقة عمل (3)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore2 من مجلد Workpapers9_2 .
2. أضف مكون جديد Label لإظهار معلومات عن التطبيق أسفل المكونات التي تم إضافتها سابقا:
   - عدّل خصائصه كالتالي:
     - سُمك الخط : FontBold
     - حجم الخط : FontSize 16
     - الظهور : Visible False
   - غير المسمى من Label2 إلى Labinfo .
3. أضف مكون Button لإظهار معلومات التطبيق عند الضغط عليه أسفل المكونات:
   - عدّل خصائصه كالتالي:
     - الارتفاع : Hight 25 pixles
     - العرض : Width 25 pixles
     - الصورة : Image about.png
     - النص : Text حذف النص الموجود.
   - غيّر المسمى من Button2 إلى BtnInfo.
4. أضف الكتل البرمجية المناسبة لمكون BtnInfo ليظهر المكون BtnInfo مع النص «Welcome To Hamad Store».
5. احفظ التطبيق باسم جديد HamadStore3 ثم اعرض التطبيق على المحاكي.

بعد التنفيذ





في وقت فراغك

استعرض مكونات واجهة الكتل البرمجية ودوّن ما تم استكشافه من مكونات فيما يلي:



عبر عن رأيــك



أوضح مفهوم أنواع الكتل البرمجية الأساسية.	☐	☐	☐
أميز بين كتل الأحداث وكتل الإجراءات، وتوظيف كل منهما في التطبيق.	☐	☐	☐
أربط المكونات (مثل الأزرار والصور) بالكتل البرمجية.	☐	☐	☐
أعرف مفهوم الحدث البرمجي ودوره في استجابة التطبيق لتفاعل المستخدم.	☐	☐	☐
أميز بين الأنواع المختلفة من كتل الأحداث.	☐	☐	☐
أفسر مفهوم الإجراء البرمجي ودوره في تنظيم الأوامر داخل التطبيق.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

الشاشات المتعددة Multiple Screens

نواتج التعلم

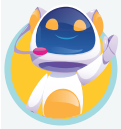
- التعرف على مميزات استخدام الشاشات المتعددة في تطبيقات الهاتف.
- شرح مميزات تعدد الشاشات من حيث التنظيم، الكفاءة، سهولة الصيانة وتطوير التطبيقات.
- التعرف على مبادئ التصميم الفعال للشاشات.
- إضافة شاشة جديدة إلى التطبيق.
- تعديل الخصائص الأساسية للشاشة مثل: السمة - المحاذاة - العنوان وغيرها.
- تصميم واجهة مستخدم بسيطة.
- تطبيق كتل برمجية للتحكم في خصائص الشاشات أثناء تشغيل التطبيق.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



معظم تطبيقات الهواتف الذكية لا تقتصر على شاشة واحدة، بل تتضمن مجموعة من الشاشات لتنظيم المحتوى، شاشة لعرض القائمة الرئيسية، وشاشة أخرى لعرض تفاصيل محددة، حيث يتفرع التطبيق لمجموعة من الشاشات الفرعية مما يجعل التطبيق أكثر ترتيباً وكفاءة.





التعلـــم

تعدد الشاشات

مميزات تعدد الشاشات

- التنظيم والترتيب: تنظيم المحتوى وترتيبه: تقسيم التطبيق إلى شاشات، بحيث تُخصص كل شاشة لمهمة محددة، مما يُسهم في تقليل الازدحام وتحسين وضوح العرض.

- الكفاءة: تحميل أسرع، إدارة أفضل للذاكرة، واستهلاك أقل لموارد الجهاز (البطارية - المعالج ...).
- سهولة الصيانة: تطوير كل شاشة بشكل مستقل، وتصحيح الأخطاء، إضافة التحديثات، دون التأثير على باقي الشاشات.
- التسلسل المنطقي للمهام، مما يسهم في تقليل التشتت بين المحتويات.
- المرونة في الاستخدام، مع انتقال سريع بين الأقسام.
- توفير الوقت، بتقليل البحث.

المبادئ الأساسية لتصميم الشاشات

تصميم الشاشات بشكل فعال في التطبيقات، يعتمد على مجموعة من المبادئ الأساسية التي تضمن سهولة الاستخدام وتناسق الواجهات وجودة تجربة المستخدم:

- التصميم الهرمي: البدء بشاشة رئيسية تعرض نظرة عامة عن وظائف التطبيق أو محتوياته ثم الانتقال إلى شاشات فرعية مخصصة لكل مهمة أو قسم، يساعد في أن يكون التنقل منظمًا، ويجب أن تكون العودة إلى الشاشة الرئيسية أو بين الشاشات واضحة وسهلة، مثل وضع زر رجوع في كل شاشة فرعية، أو أزرار تنقل ثابتة في أعلى أو أسفل الواجهة.
- التنسيق وتوحيد التصميم: استخدام قالب تصميم ثابت يضمن ثبات أماكن الأزرار، وتناسق الخطوط والألوان، وتوحيد نمط عرض البيانات في كافة الشاشات، مما يجعل التطبيق يبدو احترافيًا ويوفر على المستخدم جهد البحث عن العناصر، ويقلل نسبة حدوث أخطاء أثناء الاستخدام.
- وضوح المسارات والإرشادات: استخدام مسار واضح للمستخدم داخل التطبيق من خلال عرض اسم الشاشة الحالية، واستخدام إشارات بصرية أو تعليمات نصية لتوضيح الخيارات المتاحة، واستخدام تلميحات وقوائم منظمة تساعد المستخدم تساعد المستخدم على معرفة الموقع الحالي في التطبيق.



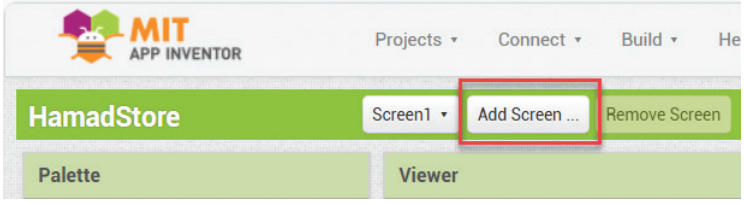
نشـــاط 1

اذكـر اثنين مـن مميـزات تعدد الشاشـات:

- .....
- .....

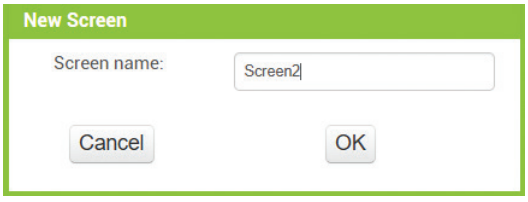
إضافة شاشة جديدة للتطبيق

- يتم إضافة شاشة جديدة للتطبيق من خلال الضغط على زر Add Screen في واجهة البرنامج.



شكل رقم (31): إضافة شاشة جديدة للتطبيق.

- تظهر الشاشة الفرعية التالية لكتابة اسم الشاشة الجديدة :



شكل رقم (32): صندوق محاورة كتابة اسم الشاشة الجديدة.

خصائص الشاشة

كل مكـون داخـل التطبيـق مثـل: زر، صـورة، مربـع نـص، ولـكل مكـوّن مجموعة مـن الخصائص يمكـن تعديلهـا لتغيير شـكله أو سـلوكه داخـل التطبيق

الشاشات المتعددة Multiple Screens

الشاشة نفسها تُعتبر مكوّنًا رئيسيًا ولها مجموعة من الخصائص والتي يمكن تعديلها حسب الحاجة مثال: تغيير لون خلفية الشاشة، وتوسيط المكونات، وتغيير أيقونة التطبيق التي تظهر عند تثبيت التطبيق على الجهاز.

الخاصية	الاستخدام
حول الشاشة (AboutScreen)	إضافة معلومات تعريفية حول التطبيق تظهر في قائمة النظام.
محاذاة أفقية (AlignHorizontal)	التحكم في ضبط محاذاة المكونات داخل الشاشة أفقيًا (يمين - يسار - وسط).
محاذاة رئيسية (AlignVertical)	التحكم في ضبط محاذاة المكونات رأسيًا (أعلى - أسفل - وسط).
لون الخلفية (BackgroundColor)	تعيين لون خلفية الشاشة.
صورة الخلفية (BackgroundImage)	إضافة صورة كخلفية للشاشة لتعزيز الجانب البصري.
أيقونة (Icon)	تحديد الأيقونة التي تُمثّل التطبيق عند تثبيته على الجهاز (تظهر فقط في الشاشة الأولى للتطبيق).
السمة (Theme)	تحديد (نمط لوني وتصميمي) للتطبيق، كاختيار تصميم كلاسيكي أو افتراضي الجهاز وهو يحدد سمة التطبيق ويوجد منها عدد من الاختيارات (Classic-Device Default- Black Title Text-Dark).
الرؤية (الظهور) (Visibility)	يُمكن استخدامها لإخفاء أو إظهار مكون. مفيدة للألعاب التي تحتوي على العديد من العناصر المتحركة التي تختفي، أو ل من الممكن استخدامها لإخفاء الترتيبات الكاملة.
قابلية التمرير (Scrollable)	قابلية تمرير محتوى الشاشة إذا تجاوز حجم محتواها أبعاد الجهاز. ملحوظة: التأكد من إلغاء تحديد خاصية «قابل للتمرير» لتعيين ارتفاع أحد المكونات (مثل ترتيب أو لوحة رسم) لملء العنصر الرئيسي.

العنوان (Title)	النص المعروض في الشريط العلوي للتطبيق. يستخدم لإعلام المستخدمين باسم التطبيق أو الشاشة الحالية. ملحوظة: لا يُمكن تغيير اسم الشاشة الأولى (Screen1) ، ولكن يُمكن تغيير عنوانها Title.

جدول (1): خصائص شاشة التطبيق

من الممكن تغيير الخصائص للشاشة وذلك من خلال واجهة المصمم Designer أو عن طريق الكتل البرمجية Blocks، مما يسمح بتصميم واجهات مرنة ومتفاعلة تتناسب مع نوع التطبيق والفئة المستهدفة من المستخدمين.



نشاط 2

■ مهمة النشاط:

تطوير التطبيق KuwaitAttractions3 من خلال تحسين تجربة المستخدم في الشاشة الأولى Screen1 لعرض معلومات عن التطبيق بحيث يتم مراعاة محاذاة المكونات داخل الشاشة وإضافة الكتل البرمجية اللازمة للانتقال إلى شاشة أخرى.

■ تنفيذ النشاط:

يمكن تنفيذ النشاط السابق باتباع الخطوات التالية:
تشغيل برنامج App Inventor ، ثم استدعاء التطبيق KuwaitAttractions3 من مجلد الأنشطة Activity.

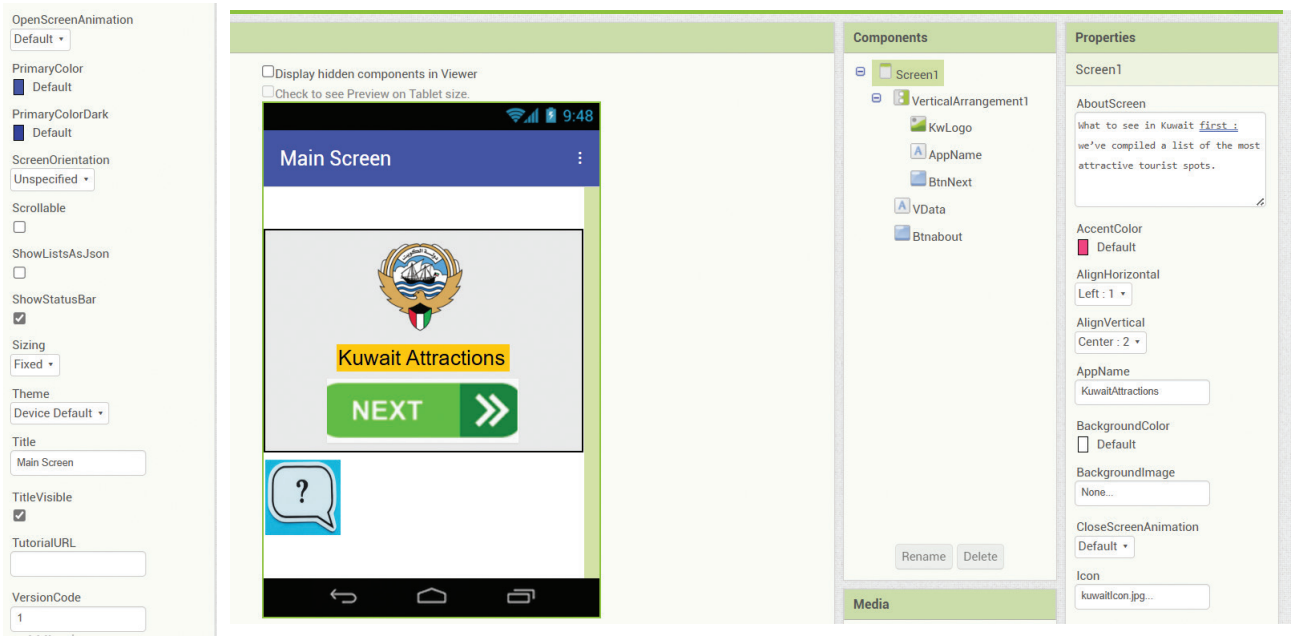
أولا : واجهة المصمم Designer:

1. اختيار الشاشة الأولى في التطبيق Screen1 من المكونات ، ثم تعديل خصائصها لتكون كالتالي:

- حول الشاشة AboutScreen: إضافة النص التالي:

What to see in Kuwait first: we have compiled a list of the most attractive tourist spots»«

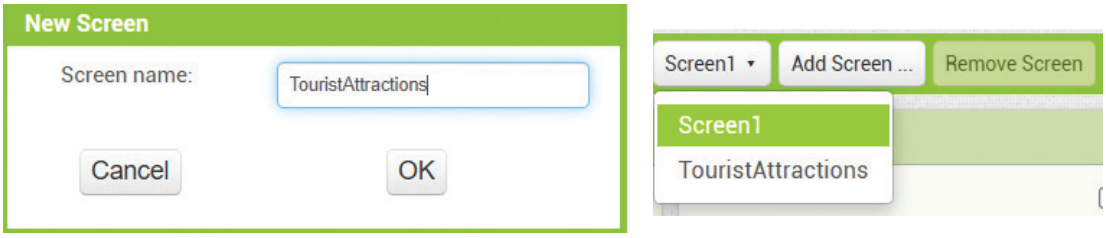
- المحاذاة الرأسية AlignVertical: وسط الشاشة center2.
- أيقونة اختيار الصورة Icon: تحميل الصورة (kuwaiticon.jpg) من مجلد Images.
- السمة Theme: اختيار (Device Default).
- العنوان Title : Main Screen.



شكل رقم (32): تعديلات خصائص الشاشة Screen1

2. إضافة شاشة جديدة للتطبيق باسم TouristAttractions:

- الضغط على زر Add Screen .
- من صندوق المحاورة New Screen يتم كتابة اسم الشاشة الجديدة TouristAttractions.
- الضغط على زر Ok.

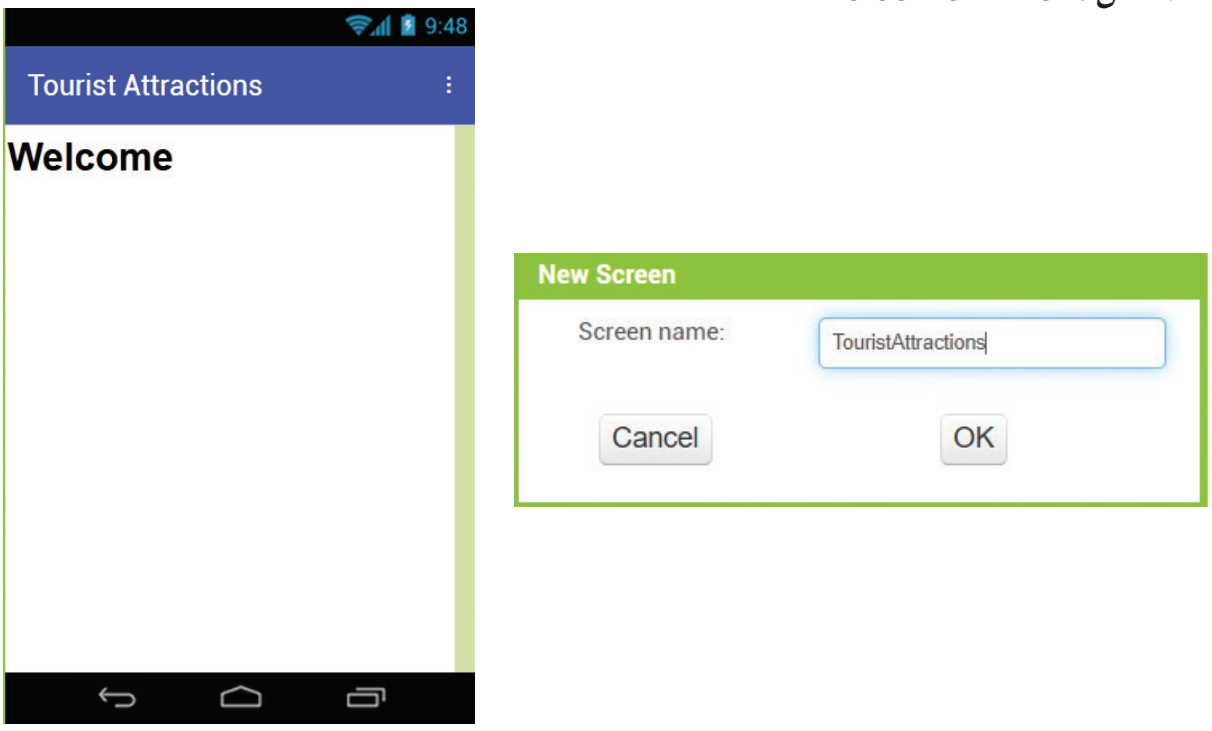


شكل رقم (33): إضافة شاشة جديدة للتطبيق باسم TouristAttractions

3. إضافة المكون Label إلى الشاشة TouristAttractions بالسحب والافلات ، ثم :

- تغيير اسم المكونLabel1 :
  - تحديد المكون Label1 منطقة Components.
  - الضغط على زر Rename لتغيير اسمه.
  - ظهور صندوق المحاورة: كتابة الاسم الجديد LabTourist.
  - في خانة New name، ثم الضغط على زر Ok.

- تغيير خصائص المكون Label1 في منطقة الخصائص Properties كالتالي:
  - FontBold : تفعيل - الخط سميك .
  - حجم الخط FontSize : 30.
  - النص Text : Welcome.

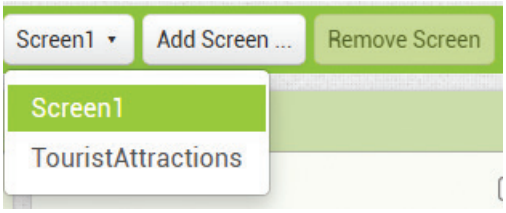


شكل رقم (34): الشاشة الجديدة TouristAttractions

لاحظ

للتأكد من إدراج الشاشة الجديدة في القائمة المنسدلة Screen1 ويمكن من خلال هذه القائمة التنقل بين شاشات التطبيق التي تم إضافتها مسبقا، كما بالشكل التالي:

- عند الضغط على السهم بجانب الاسم، تظهر جميع الشاشات المضافة .



شكل رقم (35): ظهور الشاشة الجديدة في قائمة **Screen1**

ثانيا: واجهة الكتل البرمجية Blocks :

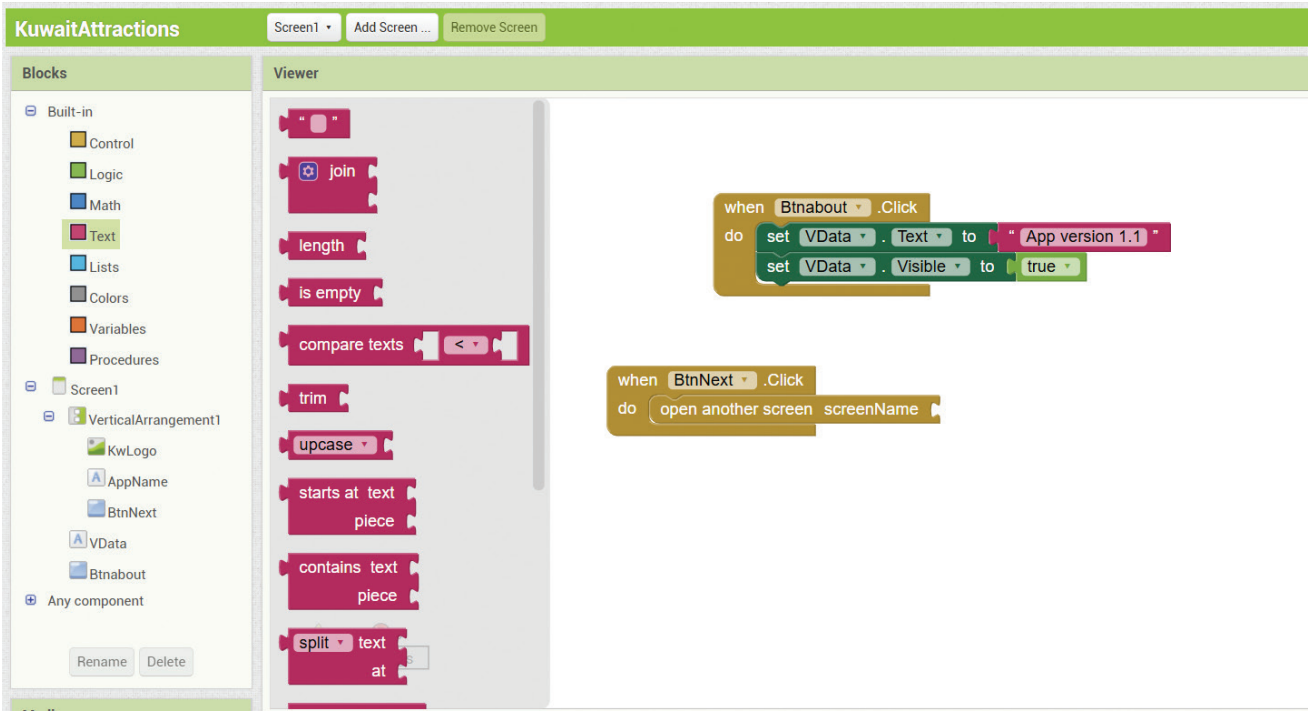
4. إضافة الكتل البرمجية المناسبة للمكون BtnNext للانتقال إلى شاشة TouristAttractions باتباع الخطوات التالية :

- الانتقال إلى منطقة الكتل البرمجية ثم اختيار الشاشة الأولى من القائمة المنسدلة وهي Screen1 .
- تحديد المكون BtnNext لتظهر الكتل البرمجية الخاصة به.
- إضافة الكتل البرمجية التالية للانتقال من (شاشة Screen1) إلى شاشة (TouristAttractions).

لاحظ

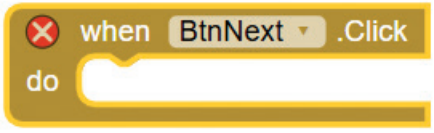
لسهولة البحث عن الكتل البرمجية:

- الضغط على أي مكان فارغ في منطقة العمل (viewer) ثم كتابة الأمر المراد البحث عنه.
- تظهر قائمة بالأوامر ، يتم اختيار الأمر المطلوب البحث عنه.
- تظهر الكتلة البرمجية الخاصة به تلقائياً على سبيل المثال إذا تم كتابة «open» ، فستظهر الكتلة البرمجية التي تمثل هذا الإجراء.



شكل رقم (36): كيفية البحث عن الكتل البرمجية بالكتابة

- اختيار الكتلة البرمجية When BtnNext.click ووضعها في منطقة العمل Viewer

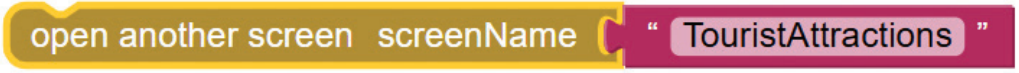


- اختيار الكتلة البرمجية open another screen screenName من تصنيف control لاستدعاء الشاشة الثانية .



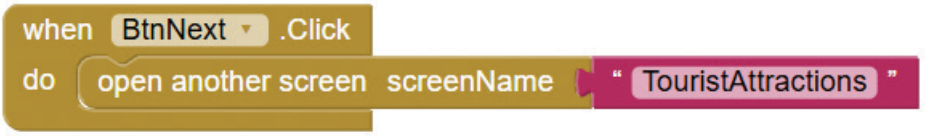
- في منطقة Blocks من Text يتم سحب الكتلة البرمجية A Text String 

- كتابة اسم الشاشة «TouristAttractions» داخل كتلة النص A Text String ليكون الشكل كما يلي:

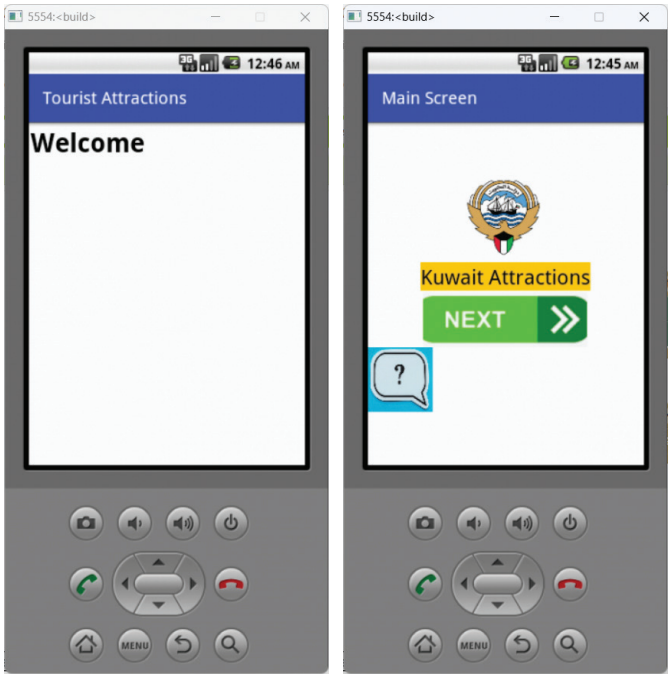




داخل كتلة النص يجب كتابة الاسم تمامًا كما تم تعريفه، بنفس الحروف الكبيرة والصغيرة حتى لا يعتبرها النظام أسماء مختلفة. بعدها يتم تجميع الكتلة البرمجية كما يلي :



5. حفظ التطبيق باسم KuwaitAttractions4 ومن ثم عرض التطبيق داخل المحاكي.



شكل رقم (37): عرض التطبيق داخل المحاكي



التطبيـــق

ورقة عمل (4)

من خلال دراستك برنامج App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore3 من مجلد أوراق العمل Workpaper9_2.
2. غير خصائص الشاشة الأولى للتطبيق Screen1 كالتالي:
   - حول الشاشة AboutScreen: أضف النص التالي:«Welcome To Hamad Store»
   - ضبط المحاذاة الرأسية AlignVertical: وسط الشاشة center2.
   - أيقونة التطبيق: Icon تحميل الصورة: HamadStoreIcon من مجلد Images.
   - السمة Theme : اختر Device Default.
   - العنوان Title :غير النص إلى Home.
3. أضف شاشة جديدة للتطبيق باسم Categories ثم أضف لهذه الشاشة التالي :
   - مكون HorizontalArrangement مع تغيير خصائصه كالتالي :
     - AlignHorizontal : Center3
     - width : Fill parent
     - داخل المكون HorizontalArrangement أضف عنصرين Button.
   - انتقل إلى الشاشة الأولى للتطبيق Screen1 ثم:
     - حدد المكون Button1.
     - أضف الكتل البرمجية اللازمة للانتقال من الشاشة الأولى Screen1 إلى شاشة Categories.
     - احفظ التطبيق باسم HamadStore4 ثم اعرض التطبيق داخل المحاكي.



في وقت فراغك

طـور التطبيـق HamadStore4 : أضـف شاشـة جديـدة وأضـف لهـا بعـض المكونـات التـي تعلمتهـا ثـم ناقشـها مـع معلمـك.



عبر عن رأيــك



أتعرف على مميزات استخدام شاشات متعددة في تطبيقات الهاتف الذكية.	☐	☐	☐
أدرج شاشة جديدة إلى التطبيق.	☐	☐	☐
أوضح مبادئ التصميم الفعال لتطبيقات متعددة الشاشات.	☐	☐	☐
أعدل على الخصائص الأساسية للشاشة : السمة - المحاذاة - العنوان.	☐	☐	☐
أطبق كتل برمجية للتحكم في خصائص الشاشات أثناء تشغيل التطبيق.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

التحكم في التدفق - الجملة الشَرطية If

نواتج التعلم

- تفسير مفهوم التحكم في التدفق البرمجي في التطبيقات الذكية.
- تمييز مكونات جملة الشرط If...Then وتوضيح دورها تنفيذ التعليمات البرمجية.
- بناء مشروع تطبيقي يعتمد على الشروط لاتخاذ القرار داخل التطبيق الذكي.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



عزيزي المتعلم يمكنك تصميم التطبيق بحيث ينفذ مجموعة من الإجراءات بناء على تحقق شروط محددة.





التعلـــم

التفكير واتخاذ القرار

الجُملة الشَرطية If في برنامج App Inventor :

تعتبر أساس للتحكم في تدفق التنفيذ داخل التطبيق، وتعتبر جزءاً جوهرياً في بناء معظم التطبيقات.

التحكم في التدفق - الجملة الشَرطية If

تستخدم الجملة الشَرطية If لاتخاذ قرارات داخل التطبيق بناءً على تحقق شرط محدد:

- إذا كان الشرط صحيحاً true: يتم تنفيذ مجموعة من التعليمات البرمجية.
- أما إذا كان الشرط غير صحيح false: يتم تجاهل التعليمات أو تنفيذ التعليمات الأخرى إذا استُخدِمَت مع else.

مكونات الجملة الشرطية If في App Inventor :

الجُملة الشَرطية If هي جزء من مجموعة الكتل البرمجية في مجموعة Control، وتتكون من:

- الكتلة if : تُستخدم للتحكم في تدفق البرنامج بناءً على تحقق شروط معينة.
- الشرط Condition : يتم تقييمه لتحديد ما إذا كان صحيحاً أو غير صحيح، مثلاً : تحقق قيمة معينة أو مقارنة بين متغيرين.
- الكتلة البرمجية Code Block : إذا تحقق الشرط Condition is true يتم تنفيذ الكتلة البرمجية الموجودة داخل الكتلة بعد then.
- إذا لم يتحقق الشرط، يمكن استخدام كتلة برمجية بعد else لتنفيذ تعليمات بديلة (اختياريه).

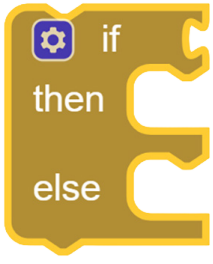
أمثلة على الجُملة الشَرطية If :

مثال 1: جُملة if-then :



- اختبار شرطاً مُحدداً :
  - إذا كان الشرط صحيحاً، يُنفّذ الإجراءات في تسلسل مُحدد من الكتل البرمجية، بعد then.
  - وإلا، يتجاهل تنفيذ الكتلة البرمجية بعد then.

مثال 2: جُملة if-then-else:



- اختبار شرطاً مُحدداً :
  - إذا كان الشرط صحيحاً: يُنفّذ الإجراءات في تسلسل الكتل البرمجية بعد then،
  - وإلا يُنفّذ الإجراءات في تسلسل الكتل البرمجية بعد else.



نشـــاط 1

مهمة النشاط:

استكمـال نشـاط الدرس السابق لتطبيق KuwaitAttractions4 لاختبار المعلومات التاريخيـة، من خـلال الانتقال إلى شاشة historicalknowledge، ثم تعديل خصائص بعض المكونات ، وبرمجـة الأزرار الخاصة بمعالم دولة الكويت، بحيث أنه عند الضغط عليه يتم التحقق من السنة المدخلة من قبـل المستخـدم، وإذا كانـت الإجابـة صحيحـة ويظهـر رسالـة: You Did Well.

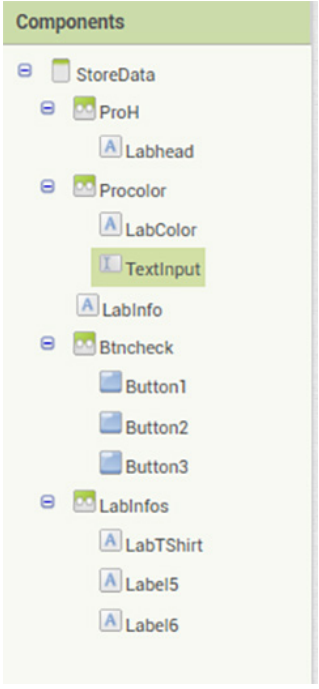
تنفيذ النشاط :

يمكن تنفيذ النشاط السابق باتباع الخطوات التالية:

- تشغيل برنامج App Inventor.

1. استدعاء التطبيق KuwaitAttractions4 من مجلد الأنشطة Activity.





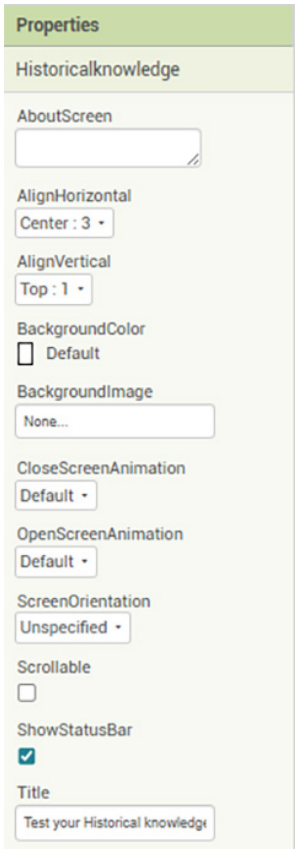
شكل رقم (38) مكونات الشاشة historicalknowledge

أولاً : واجهة المصمم Designer :

1. تحديد مكون (الشاشة) من منطقة المكونات Components، و تغيير خصائصه كالتالي:

- المحاذاة الأفقية AlignHorizontal : Center:3
- العنوان Title : Test your Historical knowledge



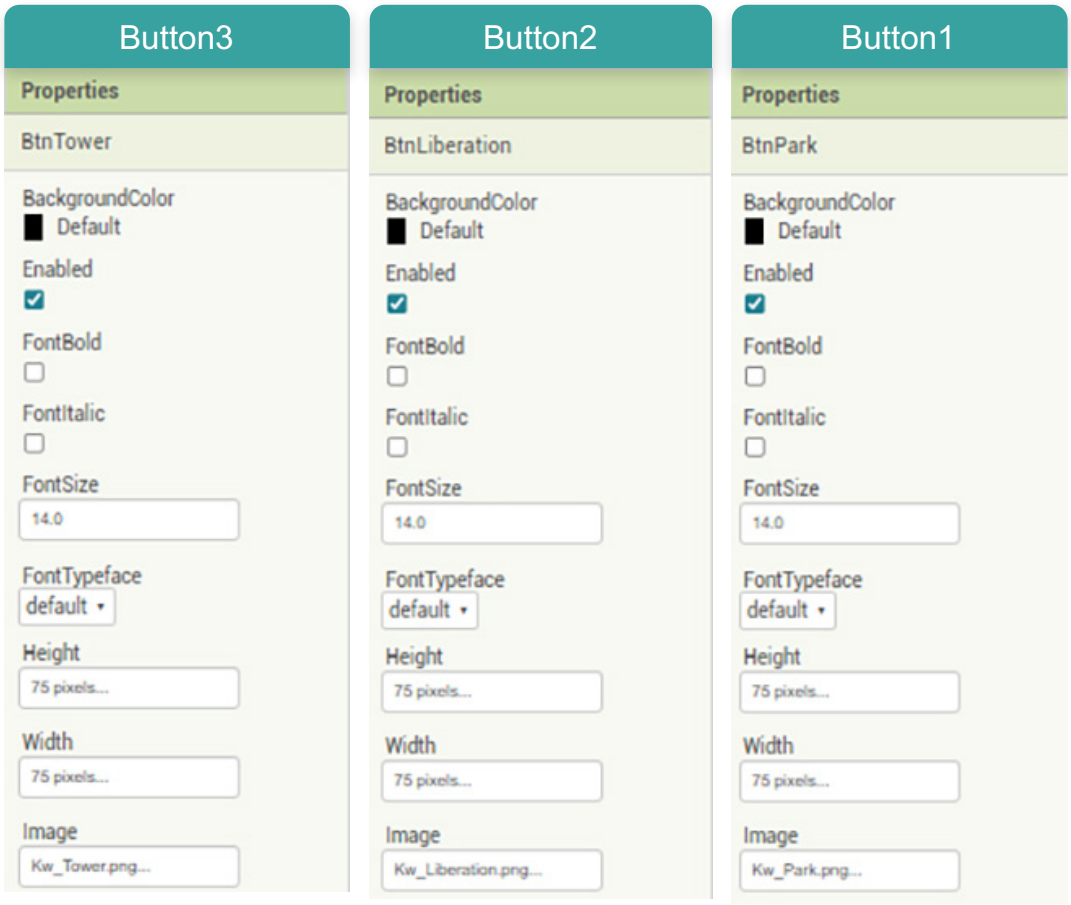


شكل رقم (39) تغيير خصائص الشاشة

2. تغيير خصائص مكونات الشاشة (الزر Button) كما يلي:

- تحديد المكون المطلوب تغيير خصائصه من خلال منطقة الخصائص Properties مع حذف النص الافتراضي بخاصية Text كالتالي :

الخاصية	المكـــــون		
	Button1	Button2	Button3
Height	75 Pixels	75 Pixels	75 Pixels
Width	75 Pixels	75 Pixels	75 Pixels
Image	Kw_Park	Kw_ Liberation	Kw_Tower
تغيير الاســـم			
Rename	BtnPark	BtnLiberation	BtnTower



3. الانتقال إلى واجهة الكتل البرمجية ثم:

- تحديد الزر BtnTower، بحيث انه عند الضغط على هذا الزر يتحقق إذا كانت السنة المدخلة في TextInput هي الإجابة صحيحة تظهر رسالة You Did Well، من خلال اتباع الخطوات التالية :



1. اختيار الكتلة البرمجية When BtnTower.click وسحبها إلى منطقة العملViewer .



2. اختيار الكتلة البرمجية if then من تصنيف Control.



3. إضافة الكتلة التالية من تصنيف الكتل الحسابية Math لمقارنة القيمة المدخلة في TextInput مع القيمة الموجودة .



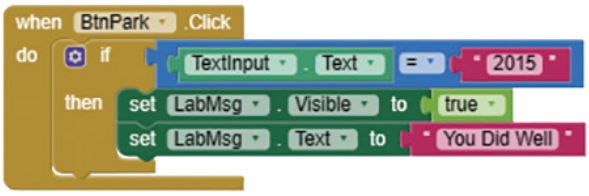
4. إضافة كتلة Text وإدخال النص حسب كل معلم سياحي.

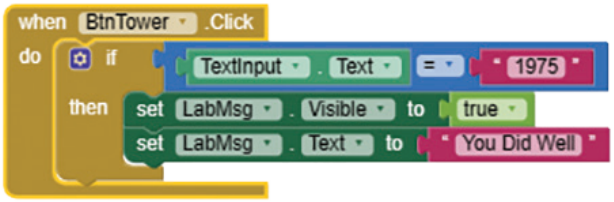
التحكم في التدفق - الجملة الشَرطية If

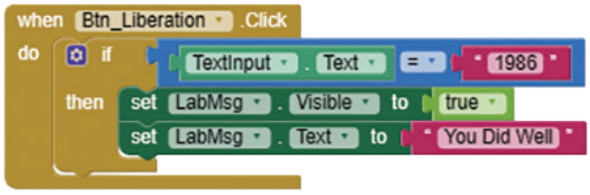


- هـل تعلم انه تم بناء أبراج الكويت في عام 1975.
- برج التحرير: تم وضع حجر الأسـاس 1986.
- حديقة الشـهيد : افتتحت المرحلة الأولى في 2015.

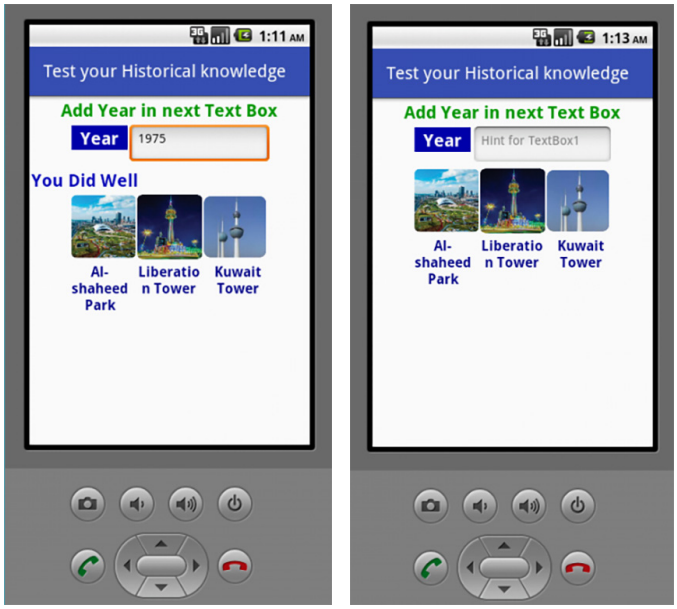
5. تجميع الكتل البرمجية لتكون كما يلي :







4. حفظ التطبيق باسم KuwaitAttractions5 وعرض التطبيق من خلال المحاكي.



شكل (40) معاينة historicalknowledge بالمحاكي .



التطبيـــق

ورقة عمل (5)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore4 من مجلد أوراق العمل Workpaper9_2.
2. انتقل إلى شاشة StoreData وغير خصائص المكونات (Button1- Button2- Button3) كالتالي:

المكون	الخصائص			
	Rename	Height	Width	Image
Button1	BtnTShirt	75 Pixels	75 Pixels	H_T Shirt
Button2	BtnCap	75 Pixels	75 Pixels	H_Cap
Button3	BtnCup	75 Pixels	75 Pixels	H_Cup

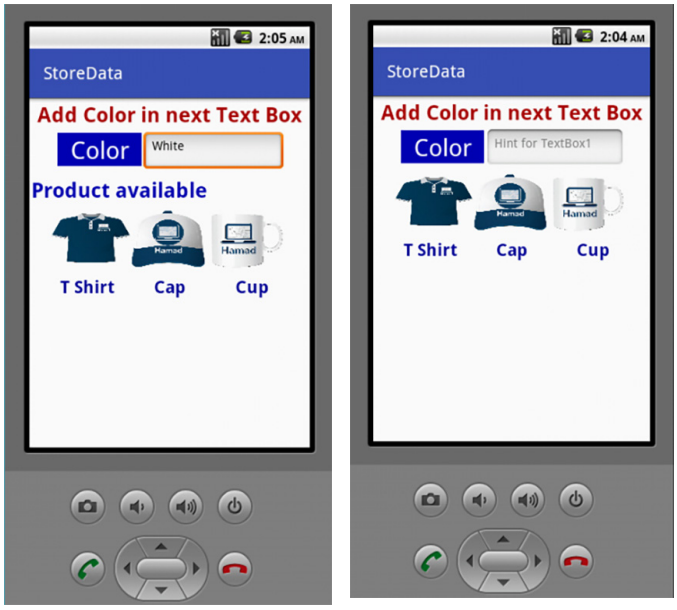
ومن خلال واجهة الكتل البرمجية Blocks نفذا ما يلي :

- أولاً : الزر BtnTShirt.
  - أضف الكتل البرمجية المناسبة للتحقق من توافر لون القميص المطلوب:
    « حيث أن المتوفر في المخزن اللون الأزرق Blue فقط ».
  - في حالة توفر المنتج تظهر رسالة Product available في LabInfo.

- ثانياً : الزر BtnCap.
  - أضف الكتل البرمجية المناسبة للتحقق من توافر لون القبعة المطلوب:
    « حيث أن المتوفر في المخزن اللون الأحمر Red فقط ».
  - في حالة توفر المنتج تظهر رسالة Product available في LabInfo.

- **ثالثاً:** الزر BtnCap.
  - أضف الكتل البرمجية المناسبة للتحقق من توافر لون الكوب المطلوب:
    "حيث أن المتوفر في المخزن اللون الأبيض White فقط":
  - في حالة توفر المنتج تظهر رسالة Product available في LabInfo.

3. حفظ التطبيق باسم HamadStore5 ثم أعرض التطبيق من خلال شاشة المحاكي.



الشاشة بعد التعديل　　الشاشة قبل التعديل



في وقت فراغك

طور من تطبيق HamadStore5، بحيث يتم إضافة الأسعار بالدينار الكويتي لكل منتج.

- ----------
- ----------
- ----------



عبر عن رأيــك



أفسر مفهوم التحكم في التدفق البرمجي في التطبيقات الذكية.	☐	☐	☐
أميز مكونات جملة الشرط If...Then وتوضيح دورها تنفيذ التعليمات البرمجية.	☐	☐	☐
أتنقل بين شاشات التطبيق.	☐	☐	☐
أعيد تسمية المكونات وأنظمها.	☐	☐	☐
أطبق كتل برمجية للتحكم في خصائص الشاشات أثناء تشغيل التطبيق.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

تابع : التحكم في التدفق - الجُملة الشَرطية If

نواتج التعلم

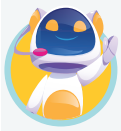
- استخدام جملة If لتنفيذ قرارات شرطية داخل التطبيقات.
- تنفيذ شروط منطقية باستخدام كتلة If في مشروع تطبيقي.
- إنشاء إجراءات تتغير حسب تحقق الشرط المحدد باستخدام If.
- تمييز الفرق بين تنفيذ التعليمات عند تحقق الشرط أو عدم تحققه عبر If.
- تطبيق جملة الشرط If..Else لحل مشكلات برمجية.
- تطبيق الشروط المركبة and - or في جملة If.
- تفسير نتيجة الشرط وتحديد المسار البرمجي المناسب.
- اختبار تنفيذ جملة If وضمان سير التطبيق حسب الشروط المحددة.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



في بعض الأحيان يحتاج التطبيق إلى اتخاذ مجموعة من الإجراءات المركبة التي تعتمد على العديد من التركيبات واتخاذ قرارات تعتمد على شروط معينة.





التعلـــم

التفكير واتخاذ القرار

■ مثال على العبارة الشرطية If

جملة if then - else if then - else :

التحكم في التدفق - الجملة الشرطية if

- آلية عمل العبارة الشرطية المركبة:
- اختبار الشرط الأول if: إذا كانت النتيجة صحيحة، يتم تنفيذ الإجراءات المحددة داخل كتلة then.
- اختبار شرط جديد else if: في حال عدم تحقق الشرط الأول، يتم الانتقال تلقائياً لاختبار شرط بديل في قسم else if.
- تنفيذ الإجراء البديل: إذا كانت نتيجة الشرط في قسم else if صحيحة، يُنفّذ الإجراءات التابعة لكتلة then الخاصة بهذا القسم.
- الحالة النهائية else: إذا لم يتحقق أي شرط من الشروط السابقة (سواء الأول أو البديل)، يتم تنفيذ الإجراءات النهائية الموجودة داخل كتلة else.

علامة «Blue Gear Sign»

علامة Blue Gear Sign في App Inventor يُطلق عليها Mutator وهي أداة تتيح للمستخدم تعديل شكل الكتلة البرمجية وإضافة أجزاء جديدة إليها:

- لا تظهر مع جملة if البسيطة، لأن هذه الجملة تعتمد على شرط واحد ولا تحتاج إلى تعديل في بنيتها الأساسية.
- تظهر مع النسخة المتقدمة من جملة الشرط، مثل: if - else و if - else if - else.
- تظهر أيضًا في بعض كتل البرمجة مثل الإجراءات Procedures أو الدوال Functions.
- تدل على أن الكتلة قابلة للتعديل بإضافة أو إزالة أجزاء منها حسب الحاجة.
- عند الضغط عليها تُفتح نافذة صغيرة تتيح إضافة مدخلات Arguments أو مكونات إضافية إلى الإجراء أو الكتلة التي تظهر عليها.
- عند الضغط عليها تُفتح نافذة لإضافة مدخلات أو مكونات، ولإغلاقها يُضغط عليها مرة أخرى أو خارج النافذة.

على سبيل المثال: في كتلة الإجراءات Procedures، يمكن استخدام زر Mutator لتوسيع الكتلة لتضمين المزيد من المعطيات Arguments عند استدعاء الإجراء.

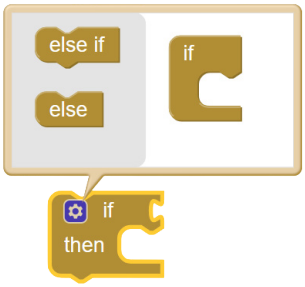
هذه الخاصية تمنح الإجراء قابلية للتخصيص، وأكثر مرونة بحيث يمكن استخدامه في أكثر من حالة عن طريق تغيير المعلومات التي يستقبلها.

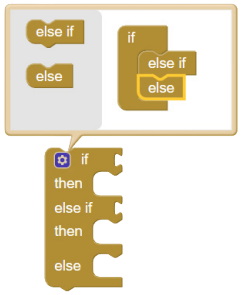
التسلسل المنطقي لبناء جملة if في App Inventor



- الانتقال إلى منطقة الكتل البرمجية Blocks.
- من تصنيف Control، يتم سحب الكتلة If...then إلى منطقة العمل.

- الضغط على العلامة Blue Gear Sign .
- ظهور قائمة منسدلة للكتلة البرمجية تحتوي على خيارات مثل: else ، else if
- اختيار المناسب من القائمة حسب الاحتياج، وسيتم تعديل الكتلة تلقائيًا لتضم الفروع الجديدة.





استخدام الشروط المركبة OR - AND مع جملة If

استخدام الشروط المركبة من خلال الكتل OR، AND وذلك للقيام بحالات التحقق من أكثر من شرط في نفس الوقت داخل جملة if.

كتلة AND (و):

تستخدم للتحقق مما إذا كانت جميع الشروط صحيحة true في نفس الوقت ، فإذا كانت كلها صحيحة، تُنفذ الأوامر.

كتلة OR (أو):

تستخدم للتحقق مما إذا كان أي من الشروط صحيح true، بحيث تكون النتيجة صحيحة فإذا تحقق أي من الشروط، تُنفذ الأوامر.

يوفر برنامج App Inventor إمكانية توسيع عدد الشروط التي تربطها بـ AND أو OR باستخدام ميزة علامة mutator الموجودة في الكتل، والتي تسمح بإضافة المزيد من المربعات لكتابة عدة شروط.



يمكن استخدام مجموعة من الشروط المركبة باستخدام OR - AND حيث :
- AND : تستخدم لربط شرطين يتطلب أن يتحققان معا.
- OR : تستخدم لربط شرطين يتحقق أحدهما فقط.

عند استخدام جمل الشروط المركبة يجب مراعاة التالي:

- وضع الشروط الأكثر تحديدا أولاً.
- التأكد من إغلاق جميع الكتل بشكل صحيح.
- اختبار جميع الاحتمالات الممكنة الحدوث.



نشـــاط

مهمة النشاط:

استكمال نشاط الدرس السابق تطبيق KuwaitAttractions5 بحيث يتم من خلال شاشة historicalknowledge تعديل الكتل البرمجية اللازمة لكل زر من أزرار المعالم السياحية للمقارنة بين القيمة المدخلة من المستخدم وتاريخ إنشاء أو افتتاح المعلم السياحي:

- إذا كانت الإجابة صحيحة تظهر رسالة You Did Well.
- إذا كانت الإجابة خطأ تظهر رسالة Try Again .

تنفيذ النشاط :

يمكن تنفيذ النشاط السابق باتباع الخطوات التالية:

- تشغيل برنامج App Inventor واستدعاء التطبيق KuwaitAttractions5 من مجلد الأنشطة Activity.
- الانتقال إلى الشاشة historicalknowledge ثم من خلال واجهة الكتل البرمجية Blocks:

أولاً: تحديد الزر BtnTower:

1. سحب الكتلة البرمجية الخاصة بالمقارنة باستخدام OR من تبويب Logic مع العلم أن الشرط المطلوب تحققه هو أن تكون القيمة المدخلة إما مساوية سنة الإنشاء أو سنة الافتتاح.



2. إضافة الكتلة البرمجية الخاصة بالتحقق من مساواة القيمة المدخلة من المستخدم والقيمة المخزنة بالتطبيق من الكتل البرمجية Math.



3. تحديد المكون TextInput ، ثم اختيار الكتلة البرمجية TextInput.Text .



4. اختيار كتلة Text (كتابة الأعوام حسب كل مَعلم)





أبراج الكويت من أبرز المعالم السياحية في البلاد، تم بناؤها عام 1975 وافتُتحت رسميًا في 1 مارس 1979تتميز بتصميمها الفريد وتُستخدم لأغراض خدمية وسياحية، وتطل على الخليج العربي بمنظرٍ رائع.

5. تكرار الخطوات 2 و 3 و 4 للتاريخ الآخر 1979.

6. لتجميع الكتل يجب تحديد العامين المطلوبين للمقارنة وهما عام 1975 و 1979، أي انه إذا أدخل المستخدم العام 1975 أو العام 1979 تكون الإجابة صحيحة.

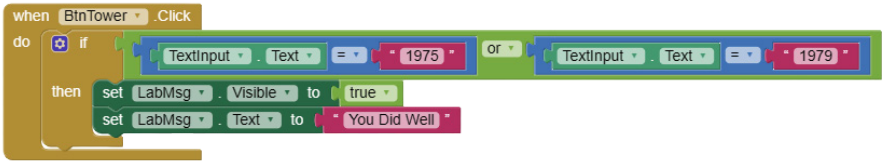
7. ترتيب الكتل البرمجية كما يلي:







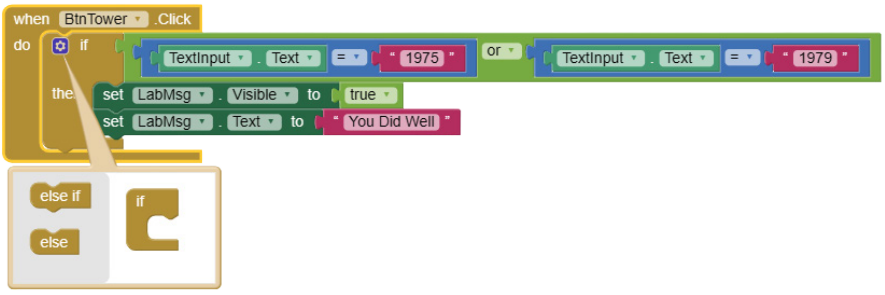
8. تجميع الكتل معا بالشكل التالي:



9. لاستكمال بناء جملة if then نحتاج الى Blue Gear Sign لإضافة جملة else ،

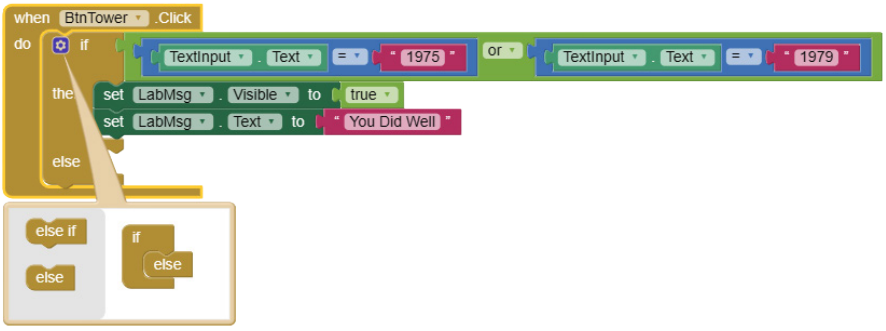
كالتالـــي:

- الضغط على Blue Gear Sign .
- اختيار else من صندوق المحاورة ووضعها داخل كتلة if .



- تظهر الكتل البرمجية كما يلي:

ظهور كلمة else في الكتلة الرئيسية لجملة if

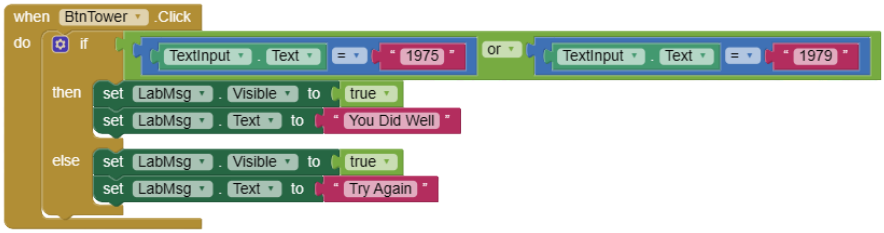


10. إضافة الرسائل الخاصة بإدخال القيم الخاطئة:





11. تجميع الكتل البرمجية لتكون كما يلي:

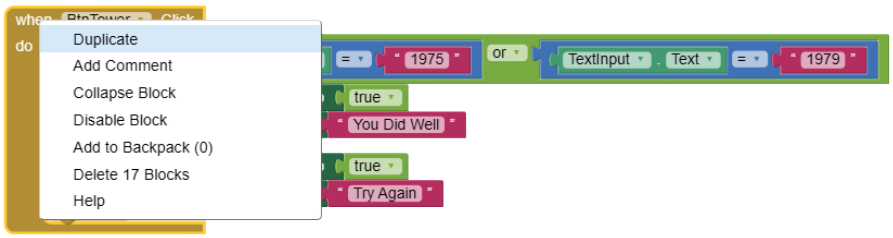


■ **ثانياً** : الزر BtnLiberation.

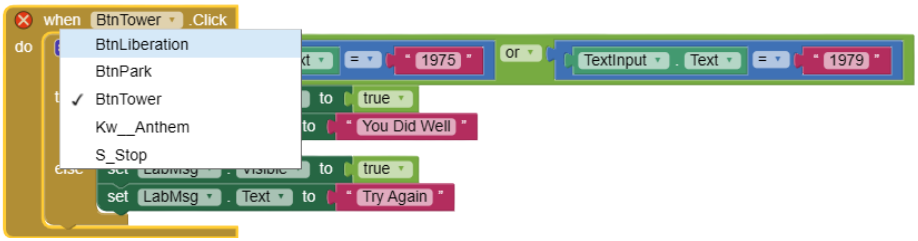


يمكنك استخدام الامر Duplicate وذلك بالضغط على الكتلة البرمجية ثم الضغط على الزر الأيمن للفأرة واختيار الأمر Duplicate .

12. إنشاء نسخة ثانية من الكتل البرمجية الموجودة مسبقًا، بهدف استخدامها مع الزر BtnLiberation مع تعديل بعض الكتل لتناسب المطلوب كما يلي:

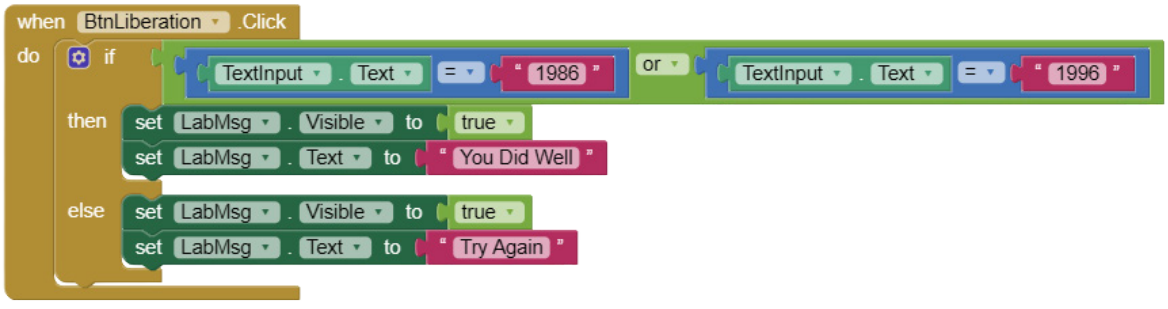


13. تعديل النسخة الجديدة حسب الحاجة، يتم هنا تغيير اسم الزر والتواريخ الخاصة بإنشاء وافتتاح برج التحرير.



يُعد برج التحرير من أبرز المعالم المعمارية في الكويت، تم وضع حجر الأساس في عام 1986، البرج يُجسد مرحلة إعادة البناء بعد التحرير و تم افتتاحه في الذكرى الخامسة لتحرير البلاد في العاشر من شهر مارس سنة 1996.

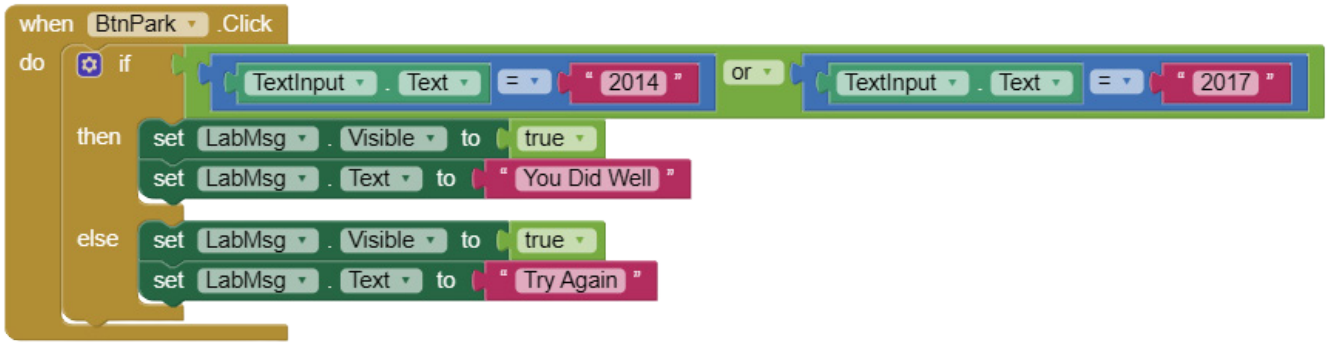
14. يصبح الشكل النهائي كما يلي:



- **ثالثاً** : زر :BtnPark.

15. إنشاء نسخة من الكتل البرمجية الموجودة مسبقًا، واستخدامها مع الزر BtnPark مع تعديل بعض الكتل لتناسب المطلوب، تغيير اسم الزر والتواريخ الخاصة بإنشاء وافتتاح حديقة الشهيد.

حديقة الشهيد هي واحدة من أكبر الحدائق الحضرية في الكويت، تقع في قلب مدينة الكويت وتجمع بين الطبيعة والتراث. افتُتحت مرحلتها الأولى في 4 مارس 2014، وتلتها المرحلة الثانية في 12 أبريل 2017، لتصبح وجهة متكاملة تضم مساحات خضراء، متاحف، ومرافق ثقافية وترفيهية تناسب جميع أفراد الأسرة.



لاحظ

- حذف الكتل الزائدة: اسحب الكتلة إلى سلة المهملات أسفل منطقة Blocks.
- تكرار الكتل: تظهر علامة تحذير بجوار الكتلة تشير إلى وجود تكرار أو خطأ منطقي، وعند الضغط عليها يعرض البرنامج رسالة توضيحية لتصحيح الخطأ.
- الكتابة في المحاكي: يظهر مربع إدخال نص مثل: TextBox، يضغط المستخدم داخله لتظهر لوحة المفاتيح، يكتب النص المطلوب ثم يضغط Done، وبعدها يضغط زر (مثل زر التحقق) لتفعيل الكتل المرتبطة بالنص.

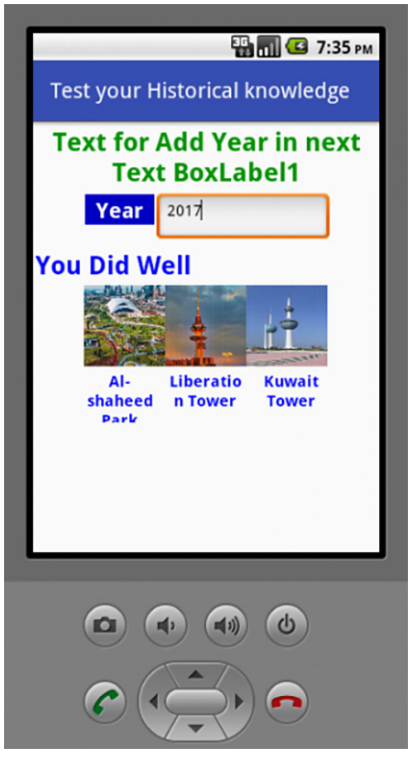
16. حفظ التطبيق باسم KuwaitAttractions6 ثم عرض التطبيق على المحاكي.

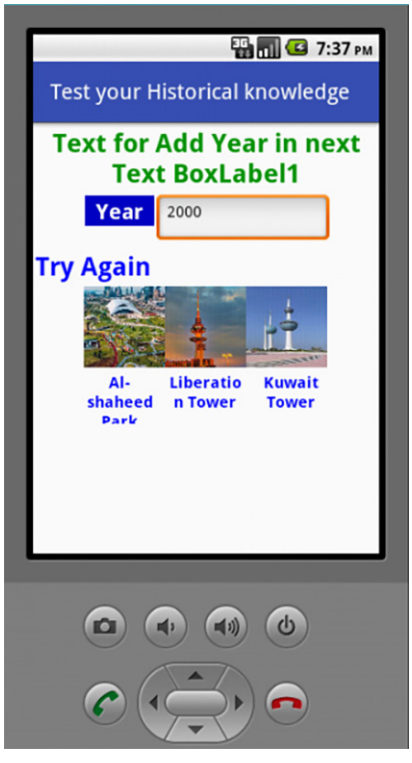


شكل (41) الكتابة داخل التطبيق في المحاكي

التطبيق عن الفتح في المحاكى	في حالة إدخال القيم الصحيحة	في حالة إدخال القيم الخاطئة







النشاط بعد التنفيذ



التطبيـــق

ورقة عمل (6)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore5 من مجلد أوراق العمل Workpaper9_2.
2. من واجهة الكتل البرمجية Blocks حدد شاشة StoreData ثم :

- **أولاً :** زر BtnTShirt.

عـدل الكتـل البرمجيـة للتحقـق ممـا إذا كانـت القيمـة المدخلـة مـن المسـتخدم (لـون المنتـج) هـي Blue أو Gray.

- إذا كان اللون متوفرًا، تظهر رسالة: Product available.
- إذا لم يكن اللون متوفرًا، تظهر رسالة: Product unavailable

- **ثانياً :** زر BtnCap.

عـدل الكتـل البرمجيـة للتحقـق ممـا إذا كانـت القيمـة المدخلـة مـن المسـتخدم (لـون المنتـج) هـي Orange أو Red.

- إذا كان اللون متوفرًا، تظهر رسالة: Product available.
- إذا لم يكن اللون متوفرًا، تظهر رسالة: Product unavailable

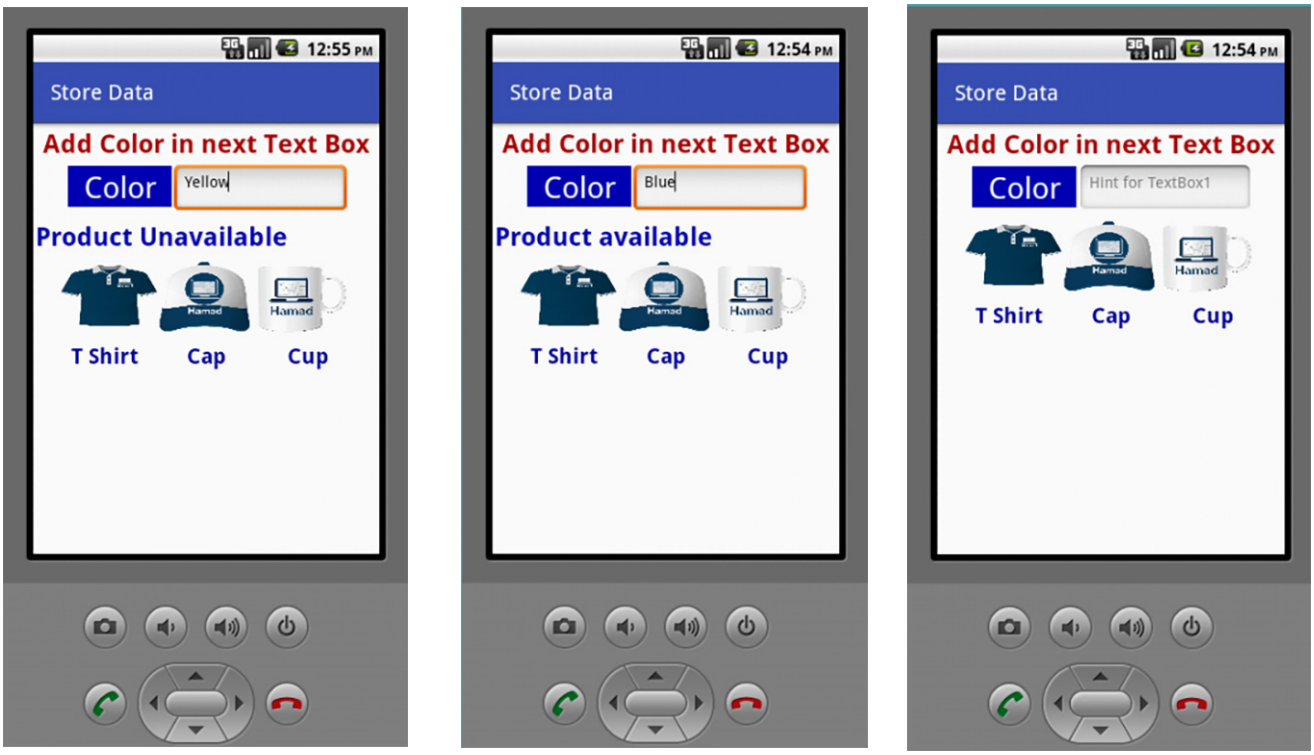
- **ثالثاً**: زر BtnCup.

عدل الكتل البرمجية للتحقق مما إذا كانت القيمة المدخلة من المستخدم (لون المنتج) هي Pink أو White.

- إذا كان اللون متوفرًا، تظهر رسالة: Product available.
- إذا لم يكن اللون متوفرًا، تظهر رسالة: Product unavailable

3. احفظ التعديلات باسم HamadStore6 ثم أعرض التطبيق من خلال شاشة المحاكي.

التطبيق عن الفتح في المحاكي	في حالة Product available	في حالة unavailable Product



التطبيق بعد التنفيذ



في وقت فراغك

استرشد بمواقع الذكاء الاصطناعي التوليدي ثم أدرج معالم إضافية من معالم دولة الكويت وطبق ما تعلمته في هذا الدرس من حيث الاستعلام عن تاريخ انشاء المعلم



عبر عن رأيــك



أستخدم جملة if لتنفيذ قرارات شرطية داخل التطبيقات.	☐	☐	☐
أنفذ شروط منطقية باستخدام كتلة if في مشروع تطبيقي.	☐	☐	☐
أنشئ إجراءات تتغير حسب تحقق الشرط المحدد باستخدام if.	☐	☐	☐
أميز بين تنفيذ التعليمات عند تحقق الشرط أو عدم تحققه عبر if.	☐	☐	☐
أطبق الشروط المركبة and - or في جملة if.	☐	☐	☐
أفسر نتيجة الشرط وتحديد المسار البرمجي المناسب.	☐	☐	☐
أختبر تنفيذ جملة if وضمان سير التطبيق حسب الشروط المحددة.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

التعامل مع النصوص والوسائط

نواتج التعلم

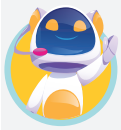
- إدراج ملفات صوتية في التطبيق باستخدام مكونات الصوت.
- ضبط مصدر الصوت من ملفات التطبيق أو من بطاقة SD أو عبر روابط URL.
- تشغيل ملف صوتي عند الضغط على زر معين في التطبيق.
- برمجة زر لإيقاف تشغيل الصوت عند الحاجة.
- اختبار تشغيل الصوت والتأكد من استجابة التطبيق عند الضغط على الأزرار.
- تنظيم ملفات الصوت داخل التطبيق لتسهيل إدارتها واستخدامها بشكل فعال.
- استخدام عدة مكونات صوت Audio Components لتشغيل ملفات متعددة.
- ربط كل زر في التطبيق بملف صوتي مختلف، بحيث يتم تشغيل الصوت المناسب عند الضغط عليه.



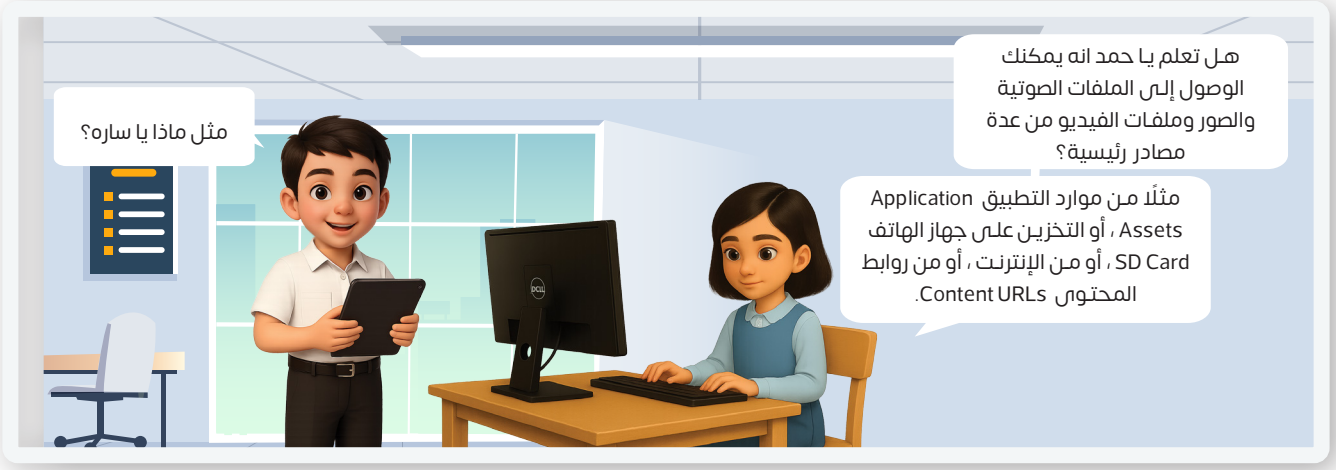
ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



- تتعامل العديد من تطبيقات الهاتف مع ملفات الصوت وملفات الصور.
- يتيح البرنامج App Inventor التعامل معهم بطريقة بسيطة وسهلة، لنتعرف عليها في هذا الدرس.





التعلم

مصادر الوسائط في App Inventor

طرق التعامل مع الصور وملفات الصوت داخل التطبيق :

تتنوع طرق التعامل مع الصور وملفات الصوت داخل التطبيق، حيث تمتاز كل طريقة بمجموعة من الخصائص وبأسلوب مختلف للوصول إليها، الجدول التالي يوضح الفرق بين هذه الطرق من حيث الخصائص وطريقة الوصول:

موارد التطبيق Application Assets

الخصائص

- ملفات مدمجة مع التطبيق.
- لا تحتاج اتصال بالإنترنت، سريعة التشغيل.
- سهلة الإعداد من قبل المصمم.

طريقة الوصول للملفات

- استخدام اسم الملف مباشرة.

العيوب

- ملفات الصوت ذات مساحة تخزين صغيرة.
- تزيد من حجم التطبيق ويصعب التعديل بعد النشر.

التخزين على الجهاز SD Card

الخصائص

- ملفات محفوظة على ذاكرة الهاتف.
- ذات مساحة تخزينية كبيرة ، مناسبة لملفات الصوت والصور كبيرة الحجم.
- مرونة في التحديث لا تحتاج إنترنت.

طريقة الوصول للملفات

- تحديد المسار المحلي للملف (اسم الملف الكامل مع الامتداد)*.

العيوب

- قد تُحذف الملفات من الجهاز.
- تحتاج إذن وصول.

الإنترنت URLs

الخصائص

- ملفات يتم تحميلها من الإنترنت.
- ذات مساحة تخزينية كبيرة، مناسبة لملفات الصوت والصور كبيرة الحجم.
- يمكن تحديث المحتوى بسهولة ولا يشغل مساحة على الجهاز.

طريقة الوصول للملفات

- رابط مباشر URL ← //:http

العيوب

- يتطلب اتصال دائم بالإنترنت.
- قد يتأخر تحميل الملفات.

روابط المحتوى Content URLs

الخصائص

- روابط مباشرة لمحتوى خارجي.
- يدعم محتوى متنوع.
- يمكن الوصول إليه من أي مكان.

طريقة الوصول للملفات

- عن طريق URL مسارات التخزين في جهاز الهاتف مثل photo gallery .

العيوب

- قد لا يعمل بدون اتصال.
- يعتمد على صلاحيات الوصول.

*يمكنك تشغيل ملف على بطاقة SD الخاصة بك عن طريق ضبط مصدر الملف إلى مُكوّن المُشغّل كما يلي :
sdcard/Music/SongName.mp3/

لاحظ

عند استخدام بطاقة الذاكرة Card SD لتخزين الملفات، يتضمن نظام Android طريقة بديلة لتصميم ملفات بطاقة الذاكرة SD كعناوين URL، يمكن إضافة بادئة file:///sdcard إلى اسم الملف، واستخدام « ترميز URL» للأحرف الخاصة، على سبيل المثال: تُستبدل دائماً بالرمز «20%»، لذا، يمكن تحديد الملف نفسه بتعيين مصدر المشغل إلى:

file:///sdcard/Music/ Song%20Name.mp3



Song%20Name.mp3
التفسير هذا هو اسم الملف الصوتي، بصيغة MP3.

file:///
التفسير بادئة URL تُستخدم للإشارة إلى أن الملف موجود محليًا على الجهاز وليس على الإنترنت.

sdcard/
التفسير يشير إلى أن الملف موجود داخل بطاقة الذاكرة أو المسار الداخلي الذي يُعامل كـ SD Card في نظام Android.

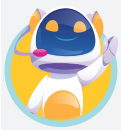
%20
التفسير هو ترميز URL للمسافة، أي أن اسم الملف الحقيقي هو: SongName.mp3.

Music/
التفسير هذا هو اسم المجلد الذي يحتوي على الملف الصوتي.

كيف يُستخدم هذا الرابط في التطبيق؟

عند استخدامه في برنامج مثل App Inventor:

- يتم تعيين هذا الرابط في خاصية Source لمكون الصوت مثل: Sound1.
- عند الضغط على زر التشغيل، يشغّل التطبيق الملف الموجود في هذا المسار.



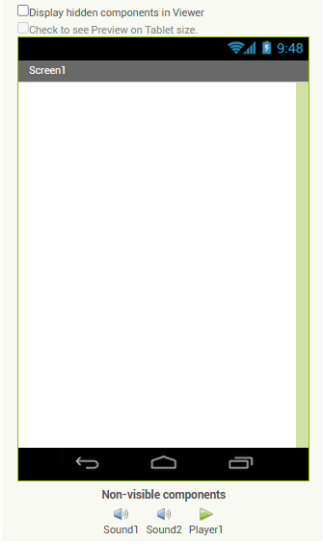
يجب أن تعلم أن App Inventor لا يتضمن (حتى الآن) أي طريقة لتخزين الملفات على بطاقة SD. كما أنه لا يتضمن (حتى الآن) طريقة لعرض الملفات الموجودة عليها. سيتعين عليك استخدام تطبيقات أخرى أو مدير ملفات هاتف Android للقيام بذلك.

الفرق بين مكون الصوت Sound ومكون المشغل Player :

- هناك فرق بين مكون الصوت Sound و مكون المشغل Player :
- مكون الصوت Sound:
  - يستخدم لتشغيل نغمة قصيرة أو صوت بسيط (مثل صوت زر أو تنبيه).
  - مناسب للملفات الصوتية الصغيرة الحجم.
  - لا يحتوي على أدوات تحكم مثل الإيقاف المؤقت أو التكرار.

- مكون المشغل Player :
  - يستخدم لتشغيل ملفات صوتية كاملة طويلة.
  - يدعم تشغيل ملفات كبيرة الحجم.
  - يحتوي على خصائص إضافية مثل: التشغيل التلقائي - التكرار - الإيقاف المؤقت والاستئناف .

- أوجه التشابه بين مكوني الصوت Sound والمشغل Player :
  - كل منهما هما مكونات غير ظاهرة non visible components (غير مرئية).
  - لا تظهر هذه المكونات داخل واجهة التطبيق للمستخدم.
  - تظهر فقط في أسفل شاشة التصميم خارج الشاشة في منطقة Viewer.



شكل (42) مكان ظهور مكوني Sound وPlayer خارج الشاشة



نشـــاط 1

مهمة النشاط:

استكمال نشاط تطبيق KuwaitAttractions6 بالانتقال من الشاشة الرئيسية إلى شاشة historicalknowledge لإدراج مكونات الصوت وإضافة الأزرار المبرمجة لتشغيل الملف الصوتي المرتبط بها عند الضغط عليها.

تنفيذ النشاط:

يمكن تنفيذ النشاط السابق باتباع الخطوات التالية:

- تشغيل برنامج App Inventor.

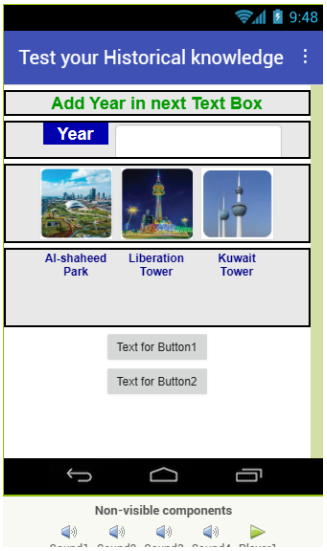
1. استدعاء التطبيق KuwaitAttractions6 من مجلد الأنشطة Activity.

- **أولاً : واجهة المصمم Designer:**

2. الانتقال إلى شاشة historicalknowledge .

3. إضافة المكونات التالية بسحبها إلى داخل شاشة التطبيق:

- Player1 - Sound1 - Sound2 - Sound3 - Sound4 - Sound5 ، من منطقة Palette ثم من Media.
- Button1 - Button2 أسفل مكونات الشاشة.



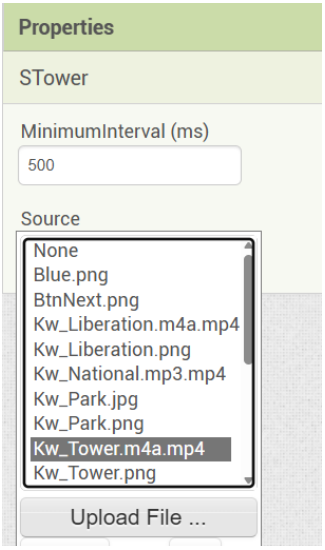
شكل (43) شاشة historicalknowledge بعد إضافة المكونات

4. تعديل خصائص المكونات السابقة وفقا للجدول التالي:

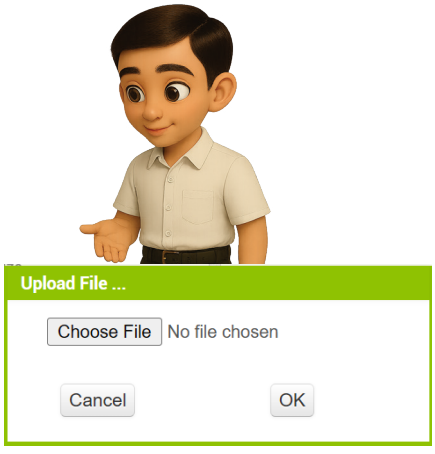
المكون	تغيير الاسم Rename	تعديل الخاصية المصدر Source
sound1	STower	Kw_Tower.m4a
Sound2	SLiberation	Kw_Liberation.m4a
Sound3	SPark	Kw_park.m4a
Sound4	SWarning	warning-alarm.mp3
Player1	SNational	Kw_National.mp3

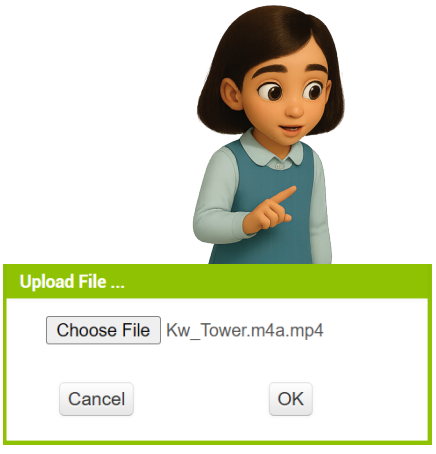
المكون	تعديل الخصائص			
	تغيير الاسم Rename	لون الخلفية BackgroundColor	النص Text	لون النص TextColor
Button1	Kw_ Anthem	Blue	National Anthem	white
Button2	S_Stop	Red	Stop Sound	white

- يتم تعديل الخاصية Source لمكون STower الذي تم إضافته في الخطوة السابقة كالتالي :
  - تحديد المكون STower.
  - الضغط على Source من صندوق المحاورة واختيار Upload flle.



- في صندوق المحاورة اختيار الملف Kw_Tower.m4a من مجلد Media.
- الضغط زر ok.

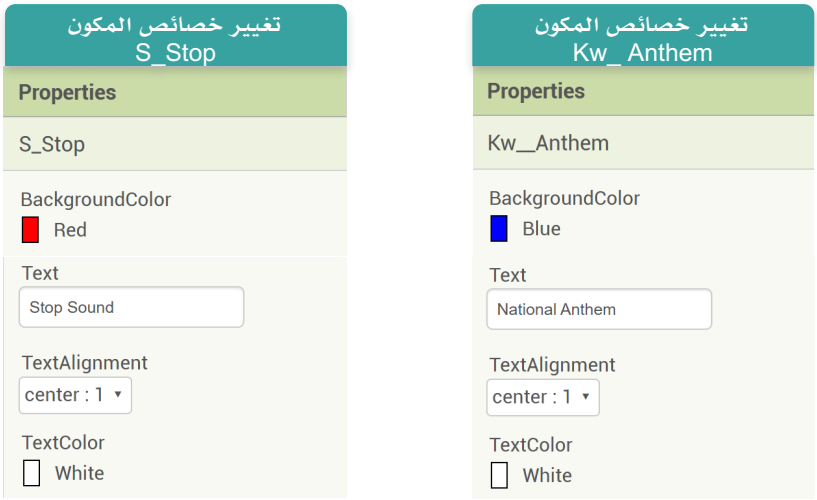




- تعديل الخاصية Source لباقي المكونات : STower- SLiberation - SPark - SNational - SWarning مع تحميل الملف الصوتي للمكونات حسب المطلوب وذلك كالتالي :



تعديل خصائص مكونات الأزرار : S_Stop - Kw_ Anthem وذلك كالتالي :





شكل (44) شاشة historicalknowledge بعد تعديل خصائص المكونات

ثانياً : واجهة الكتل البرمجية Blocks :

5. إضافة الكتل البرمجية المناسبة للأزرار التالية:

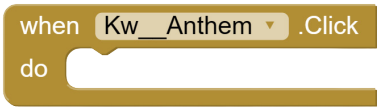
- عند الضغط على الزر Kw_ Anthem : تشغيل النشيد الوطني، وذلك باتباع الخطوات التالي :
  - تشغيل الصوت Kw_National.mp3 من خلال الكتلة البرمجية التالية:



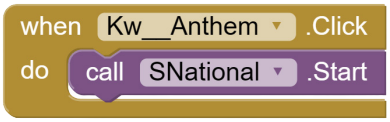
  - إيقاف الصوت من خلال الكتلة البرمجية التالية:



  - تحديد Kw__Anthem ثم إضافة الكتلة البرمجية كما يلي :



  - إضافة الكتلة البرمجية لتشغيل الصوت ليكون الشكل كالتالي:





إضافة النشيد الوطني لدولة الكويت في النشاط داخل Application Assets ويفضل في حالة إنشاء التطبيق أن يكون إضافة عن طريق رابط link داخل الكتلة البرمجية كما يلي :

set Sound2.Source to https://example.com/audio/ Kw_National.mp3

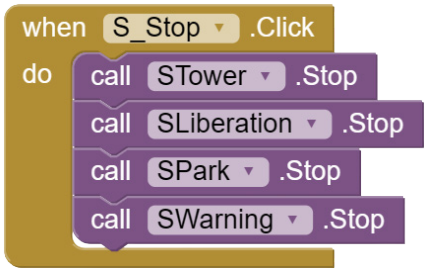
- وفي حالة الرغبة في أن يكون ملف الصوت عن طريق رابط Link يتم إضافة الكتلة التالية:



- ثم إضافة كتلة Text  ليكون الشكل النهائي كالتالي:

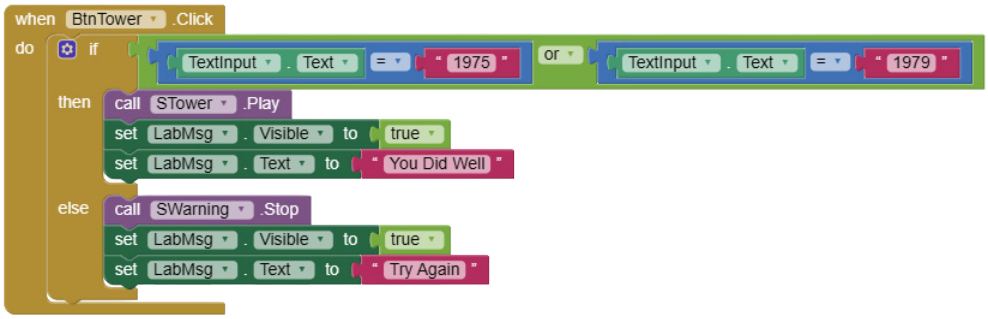


- عند الضغط على الزر S_Stop: إيقاف جميع الأصوات.

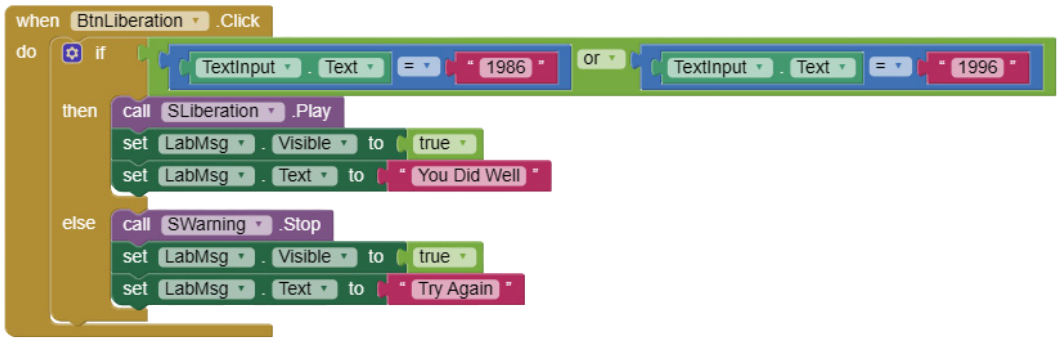


6. استكمال الكتل البرمجية للزر BtnTower ثم لباقي الأزرار، Btn_Liberation BtnPark كالتالي:

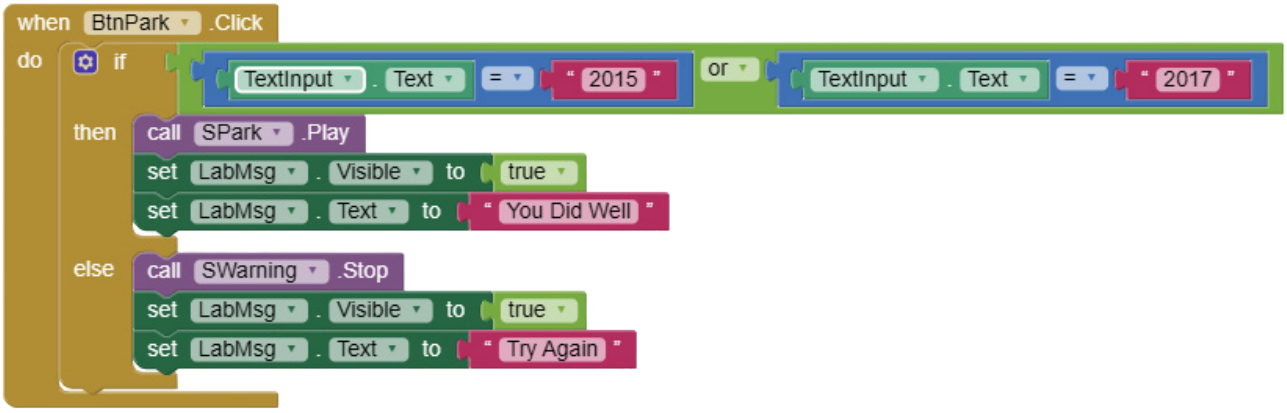
- الزر BtnTower:
  - تحديد الزر BtnTower ثم إضافة كتل تشغيل الصوت كالتالي:
    - في حال تحقق الشرط تشغيل الصوت Kw_Tower.m4a.
    - في حال عدم تحقق الشرط تشغيل الصوت warning-alarm.mp3.



- الزر BtnLiberation:
  - تحديد الزر BtnTower ثم إضافة كتل تشغيل الصوت كالتالي:
    - في حال تحقق الشرط تشغيل الصوت Kw_Liberation.m4a.
    - في حال عدم تحقق الشرط تشغيل الصوت warning-alarm.mp3.



- الزر BtnPark:
  - تحديد الزر BtnPark ثم إضافة كتل تشغيل الصوت كالتالي:
    - في حال تحقق الشرط تشغيل الصوت Kw_park.m4a.
    - في حال عدم تحقق الشرط تشغيل الصوت warning-alarm.mp3.



7. حفظ التطبيق باسم KuwaitAttractions7 وعرض التطبيق من خلال المحاكي.



التطبيـــق

ورقة عمل (7)

من خلال دراستك لبرنامج / منصة MIT App Inventor نفذ الخطوات التالية:

1. استدع التطبيق HamadStore6 من مجلد أوراق العمل Workpaper9_2.

■ أولاً: واجهة المصمم Designer:

2. حـدد شاشـة StoreData، ثـم أضـف المكونـات، وغيـر الاسـم والخصائـص لـكل مكـون وفقـاً للجـداول التاليـة :

إضافة المكونات	تغيير الاسم Rename	تعديل الخاصية المصدر Source
sound1	STShirt	H_TShirt.m4a
Sound2	SCap	H_Cap.m4a
Sound3	SCup	H_Cup.m4a
Sound4	SNotAval	Notavailable.mp3
Player1	SProudacts	H_All.mp3

المكون	تغيير الاسم Rename	تعديل الخصائص			
		لون الخلفية BackgroundColor	سُمك الخط FontBold	النص Text	لون النص TextColor
Button1	BtnMyPro	Blue	☑	All Proudcats	white
Button2	S_Stop	Red	☑	Stop Sound	white

- ثانياً: واجهة الكتل البرمجية Blocks:

3. حدد شاشة StoreData ثم نفذ التالي:

- زر BtnTShirt : عدل في الكتل البرمجية للزر كالتالي:
  - في حال توفر المنتج Product available : أضف الكتل البرمجية اللازمة لتشغيل الصوت H_TShirt.m4a.
  - في حال عدم توفر المنتج Product unavailable: أضف الكتل البرمجية اللازمة لتشغيل الصوت Not available.mp3 .
- زر BtnCap : عدل في الكتل البرمجية للزر كالتالي:
  - في حال توفر المنتج Product available : أضف الكتل البرمجية اللازمة لتشغيل الصوت H_Cap.m4a.
  - في حال عدم توفر المنتج Product unavailable: أضف الكتل البرمجية اللازمة لتشغيل الصوت Not available.mp3.
- زر BtnCup : عدل في الكتل البرمجية للزر كالتالي:
  - في حال توفر المنتج Product available : أضف الكتل البرمجية اللازمة لتشغيل الصوت H_Cup.m4a.
  - في حال عدم توفر المنتج Product unavailable: أضف الكتل البرمجية اللازمة لتشغيل الصوت Not available.mp3.

4. احفظ التعديلات باسم HamadStore7 ثم أعرض التطبيق من خلال شاشة المحاكي.



في وقت فراغك

عدّل الكتل البرمجية لتطبيق HamadAtore5 كالتالي:

- عند الضغط على الزر BtnMyPro يشغل الصوت H_All.mp3.
- عند الضغط على الزر S_Stop يوقف تشغيل الصوت H_All.mp3.



عبر عن رأيــك

أدرج ملفات صوتية في التطبيق باستخدام مكونات الصوت.	☐	☐	☐
أضبط مصدر الصوت من ملفات التطبيق أو من بطاقة SD أو عبر روابط URL .	☐	☐	☐
أربط مكونات الأزرار بمكونات الصوت لبرمجة تشغيل الصوت.	☐	☐	☐
أستطيع إيقاف تشغيل الصوت عند الحاجة.	☐	☐	☐
أختبر تشغيل الصوت مع التأكد من استجابة التطبيق عند الضغط على الأزرار.	☐	☐	☐
أنظم ملفات الصوت داخل التطبيق لتسهيل إدارتها واستخدامها بشكل فعال.	☐	☐	☐
أستخدم عدة مكونات صوت لتشغيل ملفات متعددة وربط كل منها بزر خاص به.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

المتغيرات Variables

نواتج التعلم

- التعرف على مفهوم المتغير وأهميته.
- تمييز الأنواع المختلفة من المتغيرات (عام - محلي) في السياق البرمجي.
- تحديد خطوات إنشاء المتغير وتعديله.
- استخدام المتغيرات لتنفيذ تغييرات ديناميكية في واجهة التطبيق.
- ربط الأزرار بالمتغيرات.
- اختبار سلوك المتغيرات أثناء تشغيل التطبيق.
- تطبيق المفاهيم من خلال بناء برنامج تتغير فيه خصائص الواجهة حسب المدخلات باستخدام المتغيرات.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



هل تساءلت يومًا كيف يستطيع التطبيق أن يتذكر اختيارك أو يغير سلوكه بناءً على ما تفعله؟

في درس اليوم سنكتشف معًا كيف يتم ذلك من خلال المتغيرات فهي التي تمنح التطبيقات الذكاء والمرونة.







التعلـــم

العديـد مـن التطبيقـات تحتـاج حفـظ بيانـات مؤقتـة مثـل الأسـماء، الأعمـار، أو أي معلومـات أخـرى يسـتخدمها التطبيـق أثناء التشـغيل.

من هنا يأتي دور المتغيرات Variables، فهي مثل الصناديق التي تحفظ بها القيم ليتم استخدامها داخل التطبيق.

مفهوم المتغيرات Variables

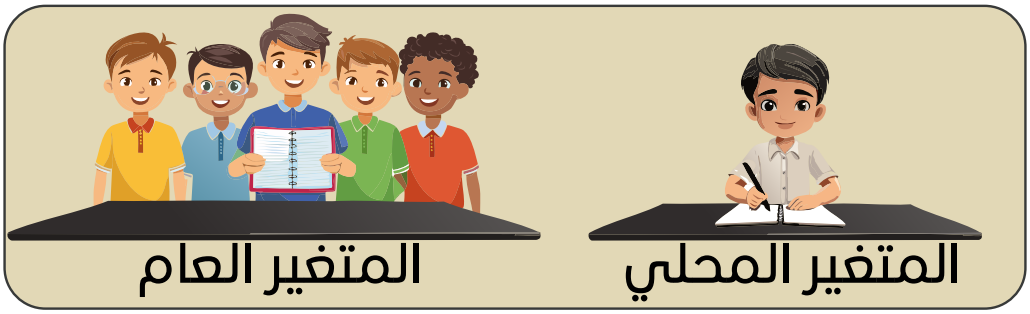
المتغير Variable هو مكان مخصص في الذاكرة لتخزين قيمة معينة مثل رقم، نص، أو نتيجة عملية حسابية، يمكن تغييرها أثناء تشغيل التطبيق، وتكمن أهميته في تسهيل التعامل مع البيانات وإعادة استخدامها بشكل ديناميكي.

أهمية المتغيرات في App Inventor

- تخزين مدخلات و بيانات المستخدم: مثل الاسم أو العمر أو النقاط.
- إدارة الحالة State: تساعد التطبيق على تذكر ما حدث سابقًا (مثل هل ضغط المستخدم على زر أم لا).
- تسهيل إجراء العمليات الحسابية: مثل حساب مجموع النقاط أو مقارنة القيم.
- إضفاء الطابع الديناميكي على التطبيق: يمكن تغيير القيم أثناء التشغيل بدلًا من أن تكون ثابتة.
- إعادة الاستخدام: بدلاً من كتابة نفس القيمة عدة مرات، يمكن استدعاء المتغير بالاسم.
- تنظيم الكتل البرمجية: تسمية المتغيرات بوضوح يجعل التطبيق أسهل في الفهم والصيانة.

أنواع المتغيرات

تخيل أنك تمتلك دفتر ملاحظات واحد يمكن للجميع القراءة منه في أي وقت، وآخر خاص بك تستخدمه وحدك في مهمة معينة؛ هكذا تمامًا يعمل المتغير العام والمتغير المحلي في البرمجة.



تنقسم المتغيرات حسب طريقة الإنشاء إلى نوعين رئيسيين:

- **المتغير العام Global Variable.**

متغير يمكن الوصول إليه في نطاقات متعددة، ويكون متاحًا لجميع الكتل البرمجية في نفس الشاشة (جميع الأحداث والإجراءات تستطيع القراءة والتغيير) هذا يعني أنه يمكن استخدام هذا المتغير، في أي مكان داخل الشاشة، للحصول على قيمته الحالية أو إعادة تعيينها إلى قيمة أخرى.

تُنشأ المتغيرات العامة باستخدام الكتلة :



- **المتغير المحلي Local Variable.**

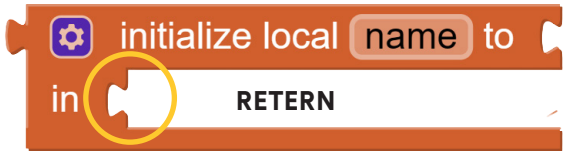
متغير مُعلن داخل دالة أو مُعامل مُمرر إليها، ويتم إنشاؤه داخل إجراء Procedure أو حدث Event أو تكرار for/while ، ولا يمكن استخدامه إلا داخل نفس الإجراء أو الحدث فقط، أي أن نطاقه محدود، وهذا يعني أنه لا يمكن الوصول إلى هذه المتغيرات إلا داخل الدالة أو الإجراء الذي أُعلن فيه أو الذي تم تمريره إليه كمعامل، ولا يكون متاحًا خارج هذا النطاق

تُنشأ المتغيرات المحلية باستخدام الكتلتين:

- الكتلة داخل جزء DO:
  - الكتلة تسمح بإنشاء متغيرات جديدة تُستخدم فقط في الإجراء في جزء DO.
  - تبدأ جميع المتغيرات بنفس القيمة في كل مرة يُشغّل فيها الإجراء.
  - لا تُرجع الكتلة قيمة، بل تنفذ تعليمات فقط.



- الكتلة داخل جزء RETURN
  - تبدأ جميع المتغيرات بنفس القيمة في كل مرة يُشغّل فيها الإجراء.
  - تختلف عن الكتلة الموضحة أعلاه: أن هذه الكتلة تُرجع قيمة محددة عند انتهاء الإجراء.
  - تحتوي على منفذ لتوصيل تعبير يُرجع نتيجة.





يمكنك إعادة تسمية المتغيرات داخل هذه الكتلة في أي وقت، وسيتم تحديث جميع الكتل الأخرى في التطبيق التي كانت تشير إلى الاسم القديم تلقائياً لتتوافق مع الاسم الجديد.

التعامل مع المتغيرات

قراءة القيمة المخزنة داخل المتغير:

تُستخدم الكتلة البرمجية get، حيث تسمح باستدعاء القيمة الحالية المخزنة في المتغير واستخدامها في العمليات الحسابية أو المنطقية أو عرضها على واجهة التطبيق.



تغيير القيمة المخزنة داخل المتغير:

تُستخدم الكتلة البرمجية set، حيث يمكن من خلالها إعادة تعيين قيمة جديدة للمتغير، وبالتالي تحديث البيانات التي يعتمد عليها التطبيق أثناء التشغيل.



جدول مقارنة بين المتغير المحلي والمتغير العام

وجه المقارنة	المتغيّر العام Global	المتغيّر المحلي Local
باستخدام الكتلة	initialize global name to	initialize local name to
مكان الإنشاء والاستخدام	متاح لجميع الكتل البرمجية في نفس الشاشة.	داخل إجراء (Procedure) أو حدث (Event) فقط.
فترة العمل بالتطبيق	طوال فترة عمل الشاشة.	أثناء تنفيذ الإجراء فقط.
يفضّل استخدامه عند	الحاجة إلى قيمة تُستخدم في أكثر من حدث أو إجراء.	الحاجة إلى قيمة مؤقتة للحساب داخل إجراء محدد فقط.

المتغيّر العام أكثر استخداماً في برنامج App Inventor، أما المتغيّر المحلي فيظهر فقط في حالات خاصة داخل إجراءات أو أحداث محددة.



نشـــاط

مهمة النشاط:

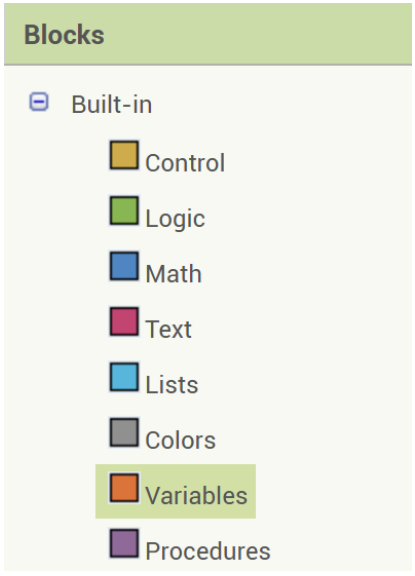
استخدام المتغيرات لتمكين المستخدم من تغيير لون خلفية الشاشة عند الضغط على الأزرار المخصصة لذلك.

تنفيذ النشاط :

يمكن تنفيذ النشاط السابق باتباع الخطوات التالية:

1. تشغيل برنامج App Inventor .
2. استدعاء التطبيق KuwaitAttractions7 من مجلد الأنشطة Activity.
3. الانتقال إلى واجهة الكتل البرمجية Blocks ثم اختيار الشاشة TouristAttractions :
4. إنشاء متغير عام Global variable باسم SelectColor:

أ من تصنيف Variables سحب الكتلة البرمجية to..... initialize global.



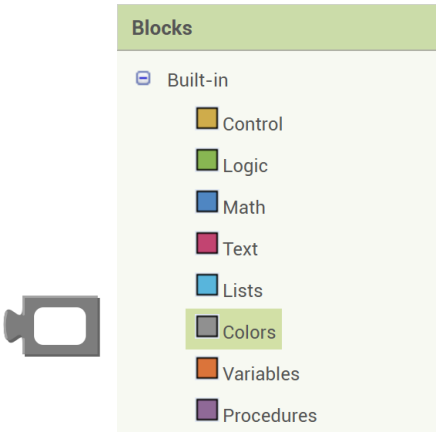


ب كتابة اسم المتغير العام SelectColor داخل الخانة name في الكتلة:



ج إضافة القيمة الافتراضية للون الشاشة ولتكن اللون الأبيض.

- من تصنيف Colors سحب اللون الأبيض.



- تصبح الكتلة البرمجية بالشكل التالي:



5. إضافة الكتل البرمجية للأزرار التالية: BtnRed - BtnBlue - BtnYellow - BtnNon لتغيير لون خلفية الشاشة عند الضغط عليها كالتالي:

أ الزر BtnRed: عند الضغط عليه تتغير خلفية الشاشة إلى اللون الأحمر:

- تحديد الزر BtnRed وإضافة الكتلة البرمجية When BtnRed.Click.



- تعيين قيمة المتغير SelectColor إلى اللون الأحمر كما يلي:
  - من تصنيف Variables، سحب الكتلة البرمجية set Variables to



  - اختيار اسم المتغير من القائمة:



- من تصنيف Colors، سحب الكتلة البرمجية الدالة على اللون الأحمر.



- تجميع الكتل البرمجية لتكون كما يلي:



- تغيير خلفية الشاشة بناء على قيمة المتغير SelectColor كالتالي:
  - تحديد الشاشة TouristAttractions
  - سحب الكتلة البرمجية set TouristAttractions.backgroundcolor



  - من تصنيف Variables، سحب الكتلة البرمجية set Variables to



  - اختيار المتغير العام من القائمة.



  - تجميع الكتل البرمجية لتكون كما يلي:



- تجميع الكتل البرمجية داخل الكتلة البرمجية When BtnRed.Click لتكون كما يلي :



ب. الزر BtnBlue: عند الضغط عليه تتغير خلفية الشاشة إلى اللون الأزرق:

- تحديد الكتلة البرمجية الكاملة للزر BtnRed وذلك بالضغط على الزر الأيمن للفارة واختيار الأمر Duplicate.
- تغيير اسم الزر وكتلة اللون كالتالي :
  - من القائمة المنسدلة اختيار اسم الزر : BtnBlue.

- سحب كتلة اللون الأزرق من تصنيف Colors وحذف الكتلة الخاصة بالزر السابق.

تصبح الكتل البرمجية للزر كما يلي:



ج الزر BtnYellow: عند الضغط عليه تتغير خلفية الشاشة إلى اللون الأصفر:

- تحديد الكتلة البرمجية الكاملة للزر BtnYellow وذلك بالضغط على الزر الأيمن للفارة واختيار الأمر Duplicate.
- تغيير اسم الزر وكتلة اللون كالتالي :
  - من القائمة المنسدلة اختيار اسم الزر : BtnYellow.
  - سحب كتلة اللون الأصفر من تصنيف Colors وحذف الكتلة الخاصة بالزر السابق.

تصبح الكتل البرمجية للزر كما يلي:



د الزر BtNon: تغيير خلفية الشاشة إلى اللون الافتراضي(الأبيض):

- تحديد الكتلة البرمجية الكاملة للزر BtNon وذلك بالضغط على الزر الأيمن للفارة واختيار الأمر Duplicate.
- تغيير اسم الزر وكتلة اللون كالتالي :
  - من القائمة المنسدلة اختيار اسم الزر : BtNon.
  - سحب كتلة اللون الأبيض من تصنيف Colors وحذف الكتلة الخاصة بالزر السابق.

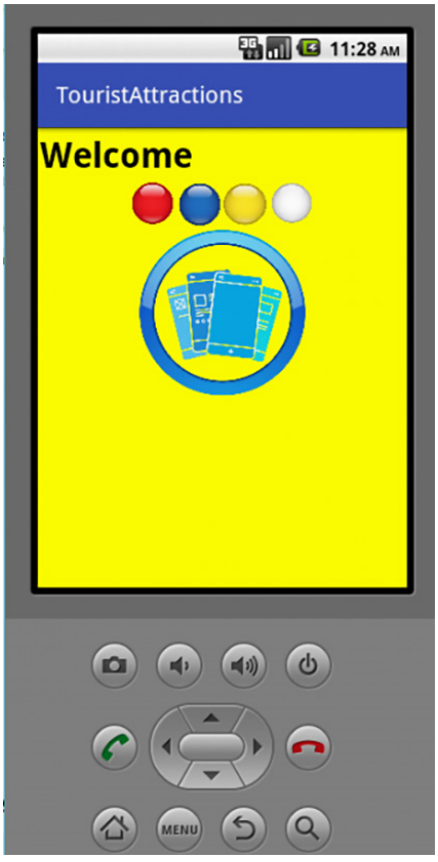
تصبح الكتل البرمجية للزر كما يلي:



6. حفظ التطبيق باسم KuwaitAttractions8 وعرض التطبيق من خلال المحاكي.



شاشة التطبيق داخل المحاكي



تغير لون خلفية شاشة التطبيق



التطبيـــق

ورقة عمل (8)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore7 من مجلد أوراق العمل Workpaper9_2.
2. انتقل إلى شاشة الكتل البرمجية ، واختر الشاشة Sale Items .
   - أنشئ متغير عام Global Variable باسم : SelectColor، وحدد له القيمة الافتراضية: اللون الأبيض.
3. أضف الكتل البرمجية اللازمة لكل زر من الأزرار الموجودة في الشاشة بحيث أنه عند الضغط على الزر:
   - BtnCyan: يتغير لون خلفية الشاشة إلى اللون السماوي Cyan.
   - BtnGreen: تغيير لون خلفية الشاشة إلى اللون الأخضر Green.
   - BtnOrange: تغيير لون خلفية الشاشة إلى اللون البرتقالي Orange.
   - BtnDefault: إعادة لون الشاشة إلى اللون الافتراضي (الأبيض) White.
4. احفظ التعديلات باسم HamadStore8 ثم أعرض التطبيق من خلال شاشة المحاكي.



شاشة التطبيق داخل المحاكي



تغير لون خلفية شاشة التطبيق



في وقت فراغك

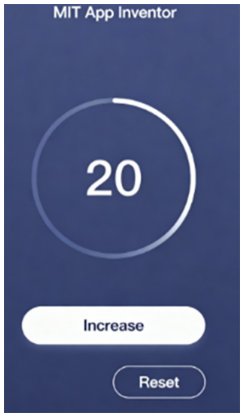
أنشئ تطبيق جديد : اسم التطبيق: عداد النقاط (Score Counter).

وصف فكرة التطبيق: تطبيق يحتوي على زرين:

- الزر الأول Button1: لزيادة النقاط.
- الزر الثاني Button2: لإعادة تعيين النقاط إلى الصفر.
- مكون Label1 : لعرض النتيجة.

خطوات التنفيذ الأساسية:

- إنشاء متغير عام باسم «score» لاستخدامه في حفظ عدد النقاط.
- إضافة زر (زيادة النقاط) وزر آخر (إعادة تعيين).
- عند النقر على زر زيادة النقاط: تحديث المتغير العام «score» بزيادة 1.
- عرض القيمة الجديدة في خانة النتيجة.
- عند النقر على زر إعادة تعيين: ضبط المتغير العام «score» إلى 0 ثم تحديث العرض.





عبر عن رأيــك



أعرف مفهوم المتغير وأهميته في البرمجة.	☐	☐	☐
أحدد خطوات إنشاء المتغير وتعديله.	☐	☐	☐
أميــز الأنــواع المختلفــة مــن المتغيــرات (العــام Global - المحلــي Local) فــي الســياق البرمجي.	☐	☐	☐
أستخدم المتغيرات لتنفيذ تغييرات ديناميكية في واجهة التطبيق.	☐	☐	☐
أربط الأزرار بالمتغيرات.	☐	☐	☐
أختبر سلوك المتغيرات أثناء تشغيل التطبيق.	☐	☐	☐
أطبق المفاهيم من خلال بناء برنامج.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

تابع : المتغيرات Variables

نواتج التعلم

- التعرف على أنواع المتغيرات حسب نوع البيانات.
- إنشاء المتغيرات لتخزين القيم الثابتة والمتغيرة.
- إنشاء متغير جديد وتعيين قيمة أولية له لاستخدامه في التطبيق.
- تحديث قيمة المتغيرات بناءً على مدخلات المستخدم .
- توظيف المتغيرات مع الشروط if لإدارة المهام المنطقية.
- تطبيق شروط if للتعامل مع النتائج المختلفة وفق قيم المتغيرات المدخلة.
- إدارة المكونات البرمجية لإظهار الرسائل والتنبيهات داخل التطبيق.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



هـل فكرت يومًـا كيف تتمكّن البرامج من حفظ المعلومات مثل عدد الزوار؟

اليـوم سنستكشـف كيـف يتـم ذلـك باسـتكمال اسـتخدام المتغيـرات Variables ونتعـرف علـى المكونـات الذكيـة التـي تجعـل التطبيقـات تفاعليـة.



التعلـــم

كمـا سـبق وذكرنـا أن المتغيـرات Variables هـي أدوات أساسـية لتخزيـن ومعالجـة البيانـات ديناميكيـاً فـي أي تطبيـق. فهـي تسـمح للتطبيـق بحفـظ المعلومـات، وتحديـث القيـم بنـاءً علـى مـدخلات المسـتخدم، وتوظيفهـا مـع الشـروط المنطقيـة if لاتخـاذ قـرارات مختلفـة وعـرض الرسـائل والتنبيهـات للمسـتخدمين ، ممـا يجعـل التطبيـق أكثـر تفاعليـة وذكاءً.

أنواع المتغيرات حسب البيانات

لكل نوع من البيانات التي نرغب في حفظها متغير خاص بها يناسب طبيعتها.

1. المتغير النصي Text Variable: لتخزين النصوص والكلمات مثل الأسماء.
2. المتغير العددي Numeric Variable: لتخزين القيم العددية مثل الأرقام.
3. المتغير المنطقي Boolean Variable: لتخزين القيم المنطقية مثل True أو False .



في App Inventor لا يتم تحديد نوع المتغير بشكل صريح عند إنشائه، بل يتحدد نوعه تلقائيًا وفقًا للقيمة المخزّنة فيه. وهذا يجعل المتغيرات مرنة وقادرة على التكيّف مع مختلف أنواع البيانات المدخلة إليها.

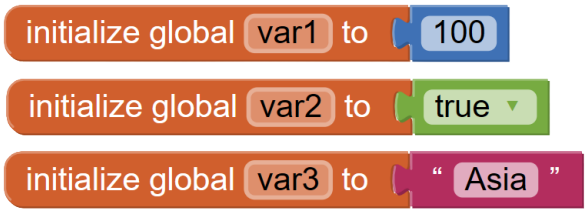
تعيين وتحديث قيم المتغيرات

تعيين القيمة الأولية عند الإنشاء (initialize)

عند إنشاء متغير جديد يجب تحديد القيمة الأولية له (Initial Value) ليبدأ بها عمله داخل التطبيق، قد تكون القيمة: رقمًا، نصًا، أو قيمة منطقية (True/False).

- يتم ذلك باستخدام الكتلة البرمجية initialize global name to ...

مثال:



تحديث القيمة أثناء التشغيل (set)

بعد تشغيل التطبيق يمكن تغيير قيمة المتغيرات وفقًا لمدخلات المستخدم أو العمليات الحسابية، حيث يتم تحديث القيمة المخزنة، بناءً على مدخلات المستخدم مثل إدخال رقم جديد في مربع نص.

- يتم ذلك باستخدام الكتلة البرمجية set global name to ...



توظيف المتغيرات والشروط

تُستخدم جملة الشرط if مع المتغيرات للتحقق من القيم المخزنة فيها، بحيث أنه عند تحقق الشرط يتم تنفيذ أوامر محددة، وعند عدم تحقق الشرط يتم تنفيذ أوامر أخرى، أي أن التطبيق يعرض النتائج المناسبة للمستخدم وفقًا لحالة الشرط.



نشـــاط

مهمة النشاط:

برمجة الزر الخاص بحجز التذاكر للتحقق من عدد التذاكر المتاحة وعرض رسالة منطقية وفقاً لذلك.

تنفيذ النشاط :

- تشغيل App Inventor واستدعاء التطبيق KuwaitAttractions8 من مجلد الأنشطة Activity.
- من واجهة الكتل البرمجية Blocks:

1. الانتقال إلى الشاشة Towertiket.
2. إنشاء المتغير العام الأول variable Global: باسم MaxVisitors لتخزين أعلى عدد للزائرين ولتكن قيمته 800.

- من تصنيف Variables سحب الكتلة initialize global name to



- تغيير الاسم في خانة name:



- من تصنيف Math سحب الكتلة Basic Number Block (الكتلة التي تحتوي على رقم)



- كتابة الرقم 800 داخل الكتلة العددية.



- تجميع الكتلة البرمجية الخاصة بالمتغير الأول.



3. إنشاء المتغير العام الثاني Global variable: باسم VisitorNum لتخزين عدد للزائرين ولتكن قيمته 760 بنفس خطوات إنشاء المتغير الأول، لتصبح الكتلة البرمجية للمتغير الثاني بالشكل التالي:



4. تحديث عدد الزوار (عبارة عن العدد المُدخل في المكون TextNum + العدد المُخزن في المتغير VisitorNum) بإضافة الكتل البرمجية اللازمة:

- سحب كتلة Set لتحديث قيمة المتغير VisitorNum بالقيمة الجديدة.



- من تصنيف Math سحب كتلة الجمع:



- سحب الكتلة الخاصة بقيمة المتغير get global VisitorNum ووضعها بالخانة الأولى لكتلة الجمع.



- سحب كتلة العدد المدخل في المكوّن TextNum.Text ووضعها بالخانة الثانية لكتلة الجمع.



- لتصبح كتلة الجمع كالتالي:

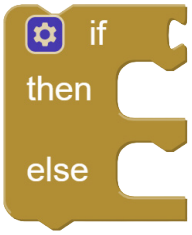


- تجميع الكتل الخاصة بتحديث عدد الزوار :

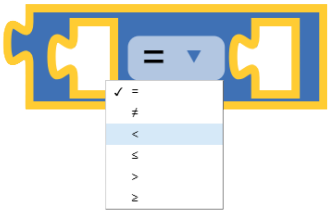


5. بدء عملية اختبار الشروط:

- إنشاء جملة شرطية if - then - else.



- من تصنيف Math سحب الكتلة الخاصة بالمقارنة واختيار رمز المقارنة المطلوب:

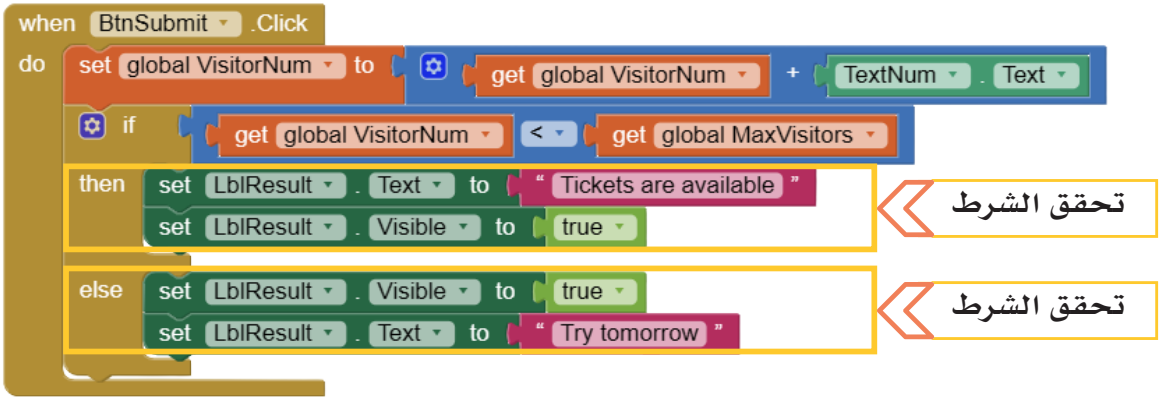


- سحب كتل قيمة المتغيرات لمقارنتها:

إذا كان (VisitorNum القيمة الجديدة < MaxVisitors):



- عند تحقق الشرط تظهر رسالة Tickets are available في مكون LblResult.
- عند عدم تحقق الشرط تظهر رسالة Try tomorrow في مكون LblResult.
- دمج الجُمل البرمجية لتعمل مع الضغط على زر BtnSubmit كما يلي:



6. حفظ التطبيق باسم KuwaitAttractions9 وعرض التطبيق من خلال المحاكي.

التطبيق عند تحقق الشرط
توفر التذاكر



التطبيق عند عدم تحقق الشرط
عدم توفر التذاكر





التطبيـــق

ورقة عمل (9)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدع تطبيق HamadStore8 من مجلد أوراق العمل.
2. انتقل إلى واجهة الكتل البرمجية Blocks .
3. طور شاشة المبيعات كالتالي:
   - حدد شاشة SaleItem.
   - إنشـي متغيـرات عامـة لتخزيـن الكميـة المتوفـرة مـن كل منتـج مـن منتجـات المتجـر:
     - QtyTShirt = 5
     - QtyCap = 8
     - QtyCup = 3
4. عند الضغط على زر المنتج: paCntB - paCntB - trihSTntB
   - إذا كانـت الكميـة المتوفـرة أكبـر مـن القيمـة المُدخلـة مـن قبـل المسـتخدم فـي (TextQty) :
     - تقل كمية المنتج بمقدار 1.
     - عرض رسالة « Purchase Successful » في المكون LabMsg.
   - وإلا :
     - عرض رسالة «Out Of Stock» في المكون LabMsg.
5. احفظ التعديلات باسم HamadStore9 ثم أعرض التطبيق من خلال شاشة المحاكي.

التطبيق عند توفر كمية المنتج



التطبيق عند عدم توفر كمية المنتج





في وقت فراغك

استكمل ما تم تنفيذه في وقت فراغك السابق للتطبيق: عداد النقاط (Score Counter) .

- أمثلة للأفكار الإضافية:
  - إضافة زر ثالث لإنقاص عدد النقاط.



عبر عن رأيــك



أتعرف على أنواع المتغيرات حسب نوع البيانات.	☐	☐	☐
أنشئ المتغيرات لتخزين القيم الثابتة والمتغيرة.	☐	☐	☐
أنشئ متغيرًا جديدًا وأعيّن له قيمة أولية لاستخدامه في التطبيق.	☐	☐	☐
أحدّث قيمة المتغيرات بناءً على مدخلات المستخدم.	☐	☐	☐
أربط المتغيرات بالمهام البرمجية.	☐	☐	☐
أوظّف المتغيرات مع الشروط (IF) لإدارة المهام المنطقية.	☐	☐	☐
أطبّق شروط if للتعامل مع النتائج المختلفة وفق قيم المتغيرات المدخلة.	☐	☐	☐
أنظم إدارة المكونات البرمجية لإظهار الرسائل والتنبيهات داخل التطبيق.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

إدارة التاريخ - دمج النصوص والقيم

نواتج التعلم

- ضبط مكون الساعة (Clock) لإظهار الوقت والتاريخ داخل واجهة التطبيق.
- استخدام المكون DatePicker لإنشاء واجهة اختيار التاريخ للمستخدم.
- توظيف الدوال البرمجية مثل MakeInstant و Duration.
- إدارة المكونات البرمجية لإظهار الرسائل والتنبيهات داخل التطبيق.
- دمج النصوص والقيم البرمجية باستخدام Join.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف



هل تساءلت يومًا كيف تستطيع تطبيقات الهواتف الذكية معرفة التاريخ والوقت أو تحديث بياناتها مع كل استخدام؟

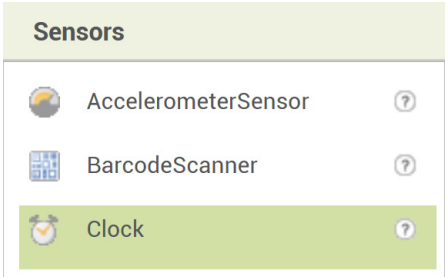
سنتعرف اليوم على المكونات الذكية التي تُحسّن تجربة تفاعل التطبيق مع المستخدم.



التعلـــم

مكوّن الساعة (Clock) في App Inventor

أحد المكونات غير المرئية (Non-visible Components)، يُستخدم بشكل أساسي لإدارة التاريخ و الوقت داخل برنامج App Inventor . يتم إضافته لشاشة التطبيق من قسم Sensors.



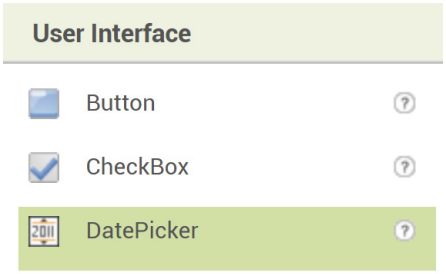
وظائفه الرئيسية:

- الحصول على الوقت والتاريخ الحاليين من جهاز المستخدم.
- ضبط مؤقت زمني (Timer) لتنفيذ أحداث برمجية في فترات زمنية محددة.
- حساب الفارق بين توقيتين أو تحويل التاريخ والوقت بين صيغ متعددة.
- إظهار الوقت الفعلي للمستخدم في التطبيق.
- جدولة أحداث تلقائية، أو إجراء الحسابات المرتبطة بالزمن.

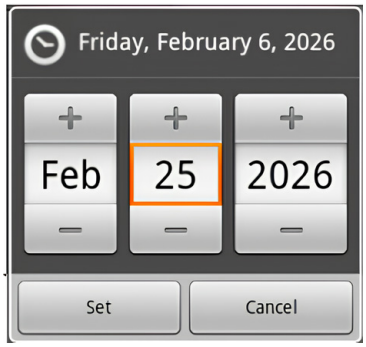
التعامل مع التاريخ في App Inventor

مكون التاريخ DatePicker

من خلاله تتم إدارة العمليات المتعلقة بالتاريخ داخل التطبيق، وتتم إضافته من قسم (User Interface).



يُستخدم لاختيار تاريخ محدد، حيث يُظهر للمستخدم عند الضغط عليه نافذة منبثقة تحتوي على تقويم ميلادي يمكن من خلاله تحديد التاريخ المطلوب.



- بعد تحديد التاريخ والضغط على زر Set، يتم تخزين القيمة التي تم اختيارها (التاريخ)، حيث أنه من الممكن استخدامه وعرضه في التطبيق.

- الكتلة البرمجية AfterDateSet هي المسؤولة عن تنفيذ الأوامر فور إغلاق نافذة التقويم.



المفاهيم والدوال البرمجية المتقدمة

دوال إنشاء اللحظة الزمنية (Instant)

- مفهوم اللحظة الزمنية (Instant)

هو كائن يمثل نقطة محددة في الزمن تشمل (السنة، الشهر، اليوم، الساعة، الدقيقة، الثانية).

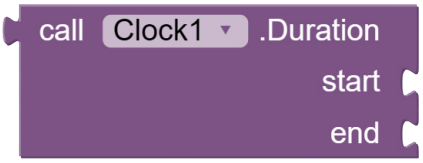
دوال الإنشاء:

- MakeInstant: إنشاء لحظة من نص أو صيغة جاهزة.
- MakeInstantFromParts: تحديد الأجزاء (سنة، شهر..) يدوياً.
- MakeInstantFromParts: إنشاء لحظة زمنية عبر تحديد الأجزاء يدويًا (سنة، شهر، يوم، ساعة، دقيقة، ثانية).

دالة المدة (Duration)

تُستخدم لحساب الفرق بين لحظتين زمنيتين، وتكون النتيجة دائماً بوحدة الملي ثانية (ms).

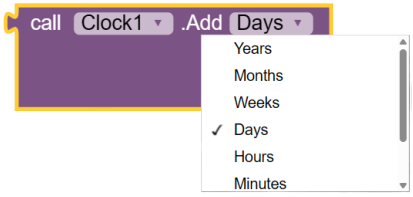
تتم قراءة ومعالجة اللحظات الزمنية بشكل افتراضي بحسب المنطقة الزمنية المحلية للجهاز، في حين يتم حساب عدد الملّي ثانية بدقة بالاعتماد على توقيت غرينتش (CTU) عند الحاجة إلى التحويل البرمجي.

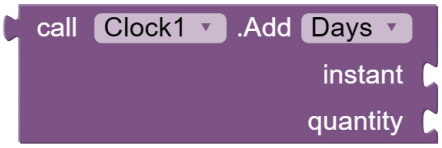


دوال الإضافة (Add)

هذه الدوال، تُستخدم لإضافة مقدار زمني إلى لحظة زمنية (Instant) موجودة، الكميات التي يمكن إضافتها باستخدام Add من خلال الاختيار من القائمة:

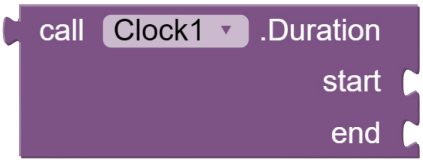
- AddSeconds : إضافة ثوانٍ إلى اللحظة الزمنية.
- AddMinutes : إضافة دقائق.
- AddHours : إضافة ساعات.
- AddDays : إضافة أيام.
- AddWeeks : إضافة أسابيع.





دوال الحساب Duration

حساب الفرق بين لحظتين زمنيتين (Instant) وتُرجع النتيجة بوحدة الملي ثانية (milliseconds).



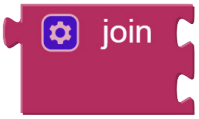
■ المدة Duration :

هي دالة تُستخدم لحساب الزمن المنقضي بين لحظتين زمنيتين، وتُعبّر هذه المدة بوحدة الملّي ثان ية.

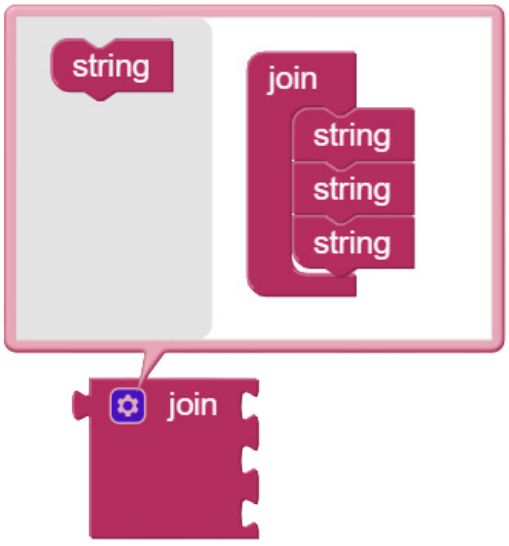
عزيزي المتعلم : تستخدم العديد من التطبيقات طرق مختلفة لدمج النصوص أو القيم معا .

معالجة النصوص باستخدام (Join)

في برنامج App Inventor تُستخدم كتلة Join الموجودة في تصنيف Text لدمج عدة نصوص أو قيم في سطر واحد.



علامة Blue Gear Sign والتي تُعرف باسم Mutator تسمح بزيادة عدد الفراغات لدمج أكثر من قيمتين.



مثال :

- دمج «Hello» مع «World» باستخدام Join يعطي النتيجة «HelloWorld».
- دمج «1» مع «2» باستخدام Join يعطي النتيجة «1+2».



يمكن إضافة سطر جديد داخل كتلة join ليتم عرض البيانات في أسطر متتالية باستخدام كتلة Text وكتابة النص \n كما يلي :





نشـــاط

مهمة النشاط :

إدارة التاريخ - دمج النصوص والقيم

استكمال نشاط الدرس السابق تطبيق KuwaitAttractions9، من خلال برمجة التطبيق ليعرض تاريخ اليوم تلقائياً عند فتح الشاشة، مع إعطاء المستخدم القدرة على تغيير التاريخ يدوياً باستخدام المكون DatePicker.

تنفيذ النشاط :

1. تشغيل برنامج App Inventor .
2. استدعاء التطبيق KuwaitAttractions9 من مجلد الأنشطة Activity.

أولاً : من واجهة المصمم Designer :

3. الانتقال إلى شاشة TicketStatus :

- إضافة المكون Datepicker من قسم User Interface على يسار المكون LblDate.
- تغيير اسم وخصائص المكون DatePicker1 كالتالي:



Rename	DateSelector
الخاصية	المطلوب
BackgroundColor	Blue
FontBold	☑
FontSize	16
Text	Select Date
TextColor	White

ثانياً : من واجهة الكتل البرمجية Blocks :

4. إظهار تاريخ اليوم مباشرة عند فتح الشاشة TicketStatus :

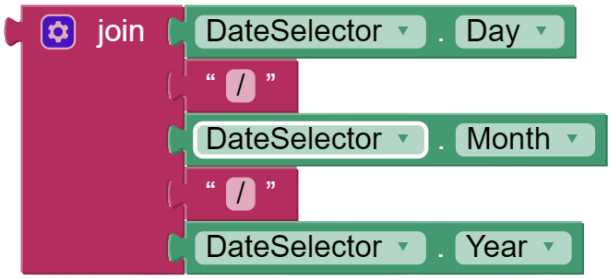
- استخدام كتلة الحدث Initialize الخاص بالشاشة.



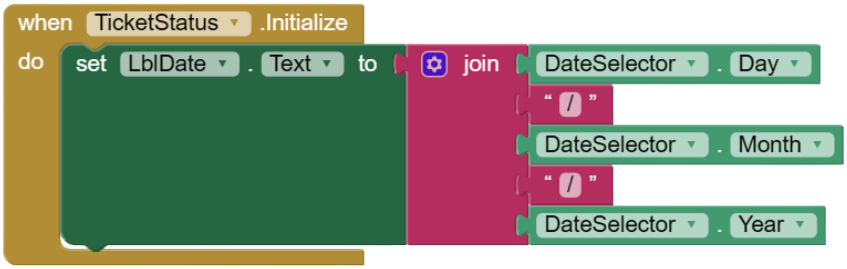
- اختيار الكتلة البرمجية اللازمة لإظهار التاريخ داخل LblDate



- إضافة الكتلة join لتجميع: اليوم و الشهر و السنة كالتالي :

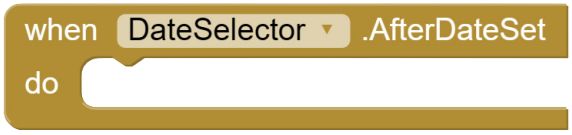


- ومن ثم تجميع الكتل معا لتكون كما يلي :

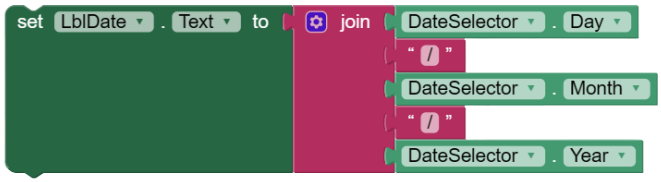


5. تحديث التاريخ الظاهر في LblDate بعد ان يختار المستخدم تاريخاً جديداً من النافذة المنبثقة عند الضغط على DateSelector:

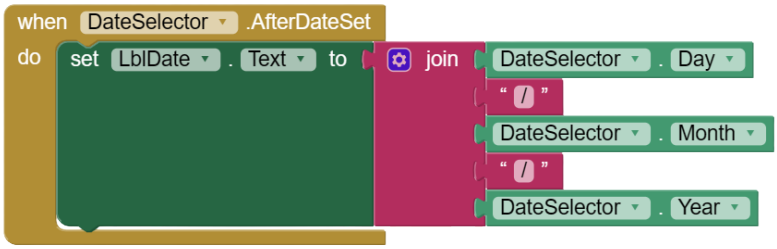
- سحب الكتلة البرمجية.



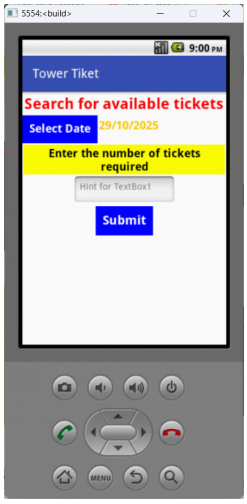
- باستخدام الأمر Duplicate يتم نسخ الكتلة البرمجية set LblDate.text to



- لتكون الكتلة كما يلي :



- حفظ التطبيق باسم KuwaitAttractions10 وعرض التطبيق من خلال المحاكي.







التطبيــق

ورقة عمل (10)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدع تطبيق HamadStore9 مجلد أوراق العمل.

- **أولا: واجهة المصمم Designer:**

2. انتقل إلى شاشة SaleItem:

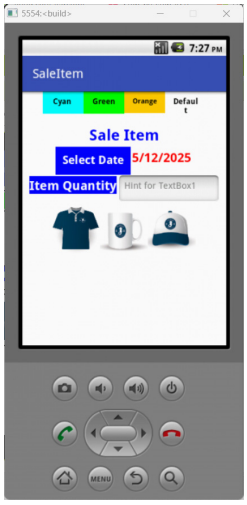
- أضف المكون DatePicker إلى واجهة الشاشة.
- غير اسم المكون البرمجي من Datepicker إلى SelectDate.
- اضبط خصائص المكون Datepicker (اللون، النص، المحاذاة) لتتطابق مع نموذج التصميم المطلوب كالتالي:

Select Date

المطلوب	الخاصية
Blue	BackgroundColor
☑	FontBold
16	FontSize
Select Date	Text
White	TextColor

- **ثانياً: واجهة الكتل البرمجية Blocks:**

3. أضف الكتل البرمجية المناسبة لإظهار تاريخ اليوم في LblDate مباشرة عند حدث فتح الشاشة SaleItem.
4. أضف الكتل البرمجية المناسبة لإظهار التاريخ الجديد في LblDate الذي حدده المستخدم عند الضغط على SelectDate.
5. احفظ التعديلات باسم HamadStore10 ثم اختبر كفاءة التطبيق من خلال شاشة المحاكي.







في وقت فراغك

بناء تطبيق «حاسبة العد التنازلي للأحداث»

- **المكونات (Designer)**
  - DatePicker و TimePicker: لاختيار التاريخ والوقت المستهدف للحدث.
  - Clock: للحصول على الوقت الحالي من جهاز المستخدم وتفعيل المؤقت.
  - Label: لعرض النتيجة النهائية للعد التنازلي.
- **البرمجة (Blocks)**
  - الدالة MakeInstant: لدمج قيم التاريخ والوقت المختارين وتحويلهما إلى نقطة زمنية محددة.
  - الدالة Duratio: لحساب الفارق الزمني (المدة) بين اللحظة المختارة واللحظة الحالية.
  - كتلة Join: لتنسيق الأرقام مع النصوص (أيام، ساعات، دقائق) لعرض النتيجة بشكل مقروء.
  - حدث Clock.Timer: لتحديث عملية الحساب وعرضها في الملصق بشكل تلقائي ومستمر.



عبر عن رأيــك



أضبط مكون الساعة (Clock) لإظهارالتاريخ في التطبيق.	☐	☐	☐
أستخدم المكون DatePicker لإنشاء واجهة اختيار التاريخ.	☐	☐	☐
أوظف الدوال البرمجية مثل MakeInstant - Duration.	☐	☐	☐
أنظم إدارة المكونات البرمجية لإظهار الرسائل.	☐	☐	☐
أدمج النصوص والقيم البرمجية باستخدام Join.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

التعامل مع القوائم - الفهرس

نواتج التعلم

- شرح مفهوم القوائم Lists كهيكل بيانات منظم.
- إنشاء قائمة بيانات List باستخدام الكتلة make a list.
- زيادة عدد العناصر داخل كتلة القائمة list من خلال علامة Mutator.
- إضافة عناصر جديدة للقائمة باستخدام الكتلة add items to list.
- التعرف على المكوّن ListPicker.
- استخدام الفهرس Index للوصول إلى عناصر القوائم في App Inventor.
- اختبار تفاعل القوائم عبر المحاكي للتأكد من صحة العمل.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

القوائم Lists

طريقة منظمة لتخزين مجموعة من القيم تحت اسم واحد، وتعتبر القوائم أحد أنواع هياكل البيانات، من خلالها يمكن استرجاع البيانات ومعالجتها.

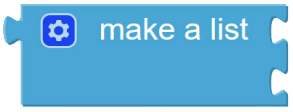
إنشاء قائمة Lists

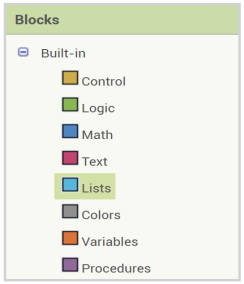
لإنشاء قائمة List في برنامج App Inventor وإدارتها وتخزين البيانات بداخلها ، يتم اتباع الخطوات التالية:

من منطقة الكتل البرمجية Blocks:

1. إنشاء القائمة:

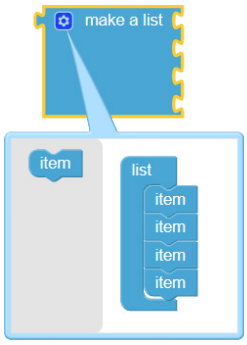
سحب كتلة make a list من تصنيف Lists.





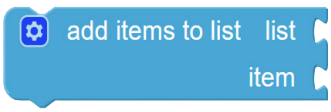
2. زيادة عدد العناصر داخل كتلة القائمة list:

استخدام علامة Blue Gear Sign (Mutator) لزيادة عدد الفراغات داخل الكتلة لتناسب عدد عناصر القائمة المطلوب إدخالها.



3. توسيع القائمة:

إضافة عناصر جديدة إلى قائمة موجودة مسبقاً أثناء تشغيل التطبيق باستخدام الكتلة add items to list.

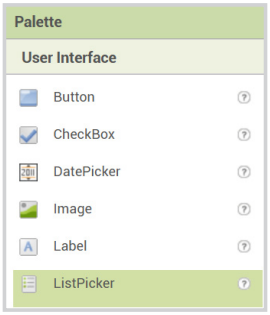


المكوّن ListPicker

مكوّن مرئي يتم إضافته للشاشة من تصنيف User Interface حيث يظهر في واجهة المصمم (Designer) كأنه زر (Button)، ولكنه برمجياً يعمل كنافذة عرض.

- عند الضغط عليه يعرض قائمة من الخيارات المُخزنة فيه مثل:
  - أسماء.
  - منتجات.
  - مدن ...

يسمح للمستخدم باختيار عنصر واحد فقط من القائمة المعروضة.



خصائص المكون ListPicker:

يمكن التحكم في مظهر ListPicker من خلال تعديل خصائصه المرئية (Properties) التي تؤثر على شكل المكون والقائمة، مثل:

- محاذاة النص .
- لون الخلفية.
- لون الخط.
- حجم الخط وغيرها.



حيث تساعد هذه الخصائص على تصميم واجهة أكثر وضوحاً وجاذبية للمستخدم داخل التطبيق.

الفهرس Index

استخدام الفهرس Index داخل القوائم للوصول إلى عناصر القوائم في App Inventor، يبدأ ترقيم عناصر القائمة من الرقم (1)، أي أن العنصر الأول في القائمة يحمل الفهرس 1 دائماً.

الوصول للعنصر باستخدام الفهرس:

يمكن الوصول السريع إلى أي عنصر داخل القائمة عند معرفة:

- رقم الفهرس (Index) الخاص بالعنصر.
- اسم القائمة التي ينتمي إليها.

مثال: استرجاع العنصر الثاني



توظيف الفهرس في:

- البحث داخل القوائم.
- عرض التفاصيل المرتبطة بعنصر معين.
- معالجة البيانات مثل التعديل، الاسترجاع، أو الربط بقائمة أخرى.



النشـــاط

مهمة النشاط:

استكمال تطبيق KuwaitAttractions10 وذلك بإعداد شاشة VisitorRecord لعرض أنواع الزيارة، وإعداد القوائم والمتغيّرات اللازمة لتخزين بيانات الزائرين والمواعيد، ثم تشغيل التطبيق للتحقق من تحميل الخيارات بشكل صحيح داخل المحاكي.

تنفيذ النشاط:

1. تشغيل برنامج App Inventor.

أولاً: واجهة المصمم Design.

2. الانتقال إلى شاشة VisitorRecord:

- إضافة المكون ListPicker إلى الشاشة بالمكان المناسب.
- تغييراسمه Rename إلى: VisitList.
- تغيير خصائصه التالية:



الخاصية	المطلوب
BackgroundColor	Blue
FontBold	☑
FontSize	18
Text	Visit Type
TextAlignment	Center:1
TextColor	White
Title	Choose Visit Type

ثانياً: واجهة الكتل البرمجية Blocks.

- **المرحلة الأولى: إعداد قائمة بأنواع الزيارة**

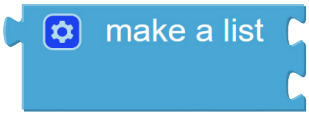
3. إنشاء المتغير العام ServicesList: الذي يمثل أنواع الزيارة لأبراج الكويت:

(Orther — School Trip — Toursit Group — Visit Single)

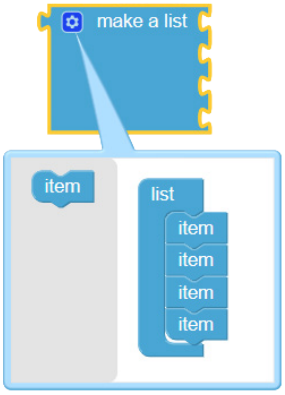


4. إنشاء List لإضافة أنواع الزيارة من خلال :

- اختيار الكتلة Make a list من تصنيف List.



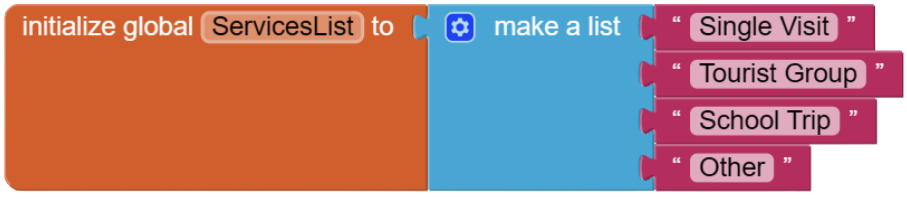
- استخدام (Mutator Sign Gear Blue) لإضافة أربع عناصر تمثّل أنواع الزيارة.



- استخدام كتلة النص text لكتابة أنواع الزيارات.



- تجميع الكتل لتكوين قائمة مكتملة بأنواع الزيارة.



■ **المرحلة الثانية:** إنشاء قوائم فارغة

- إنشاء قوائم فارغة لاستقبال مدخلات المستخدم لاحقاً ولضمان عدم وجود بيانات قديمة.
- استخدام الكتلة create empty list لتكون القائمة فارغة عند الإنشاء، حيث ستُضاف البيانات لاحقاً من قبل المستخدم عند تسجيل الزيارة .



5. إنشاء المتغير العام AppointmentDetails: لتخزين تفاصيل مواعيد الزيارة (القائمة فارغة).



6. إنشاء المتغير العام AppointmentsList: لتخزين بيانات الزيارات (القائمة فارغة).



7. إنشاء المتغير العام VisitorList: لتخزين بيانات الزائرين (القائمة فارغة).



■ **المرحلة الثالثة:** تحميل بيانات أنواع الزيارة عند فتح الشاشة

8. عند تهيئة شاشة VisitorRecord يتم تحميل جميع أنواع الزيارات داخل المكون VisitList لتكون جاهزة للاختيار من قبل المستخدم:

- استخدام كتلة الحدث.



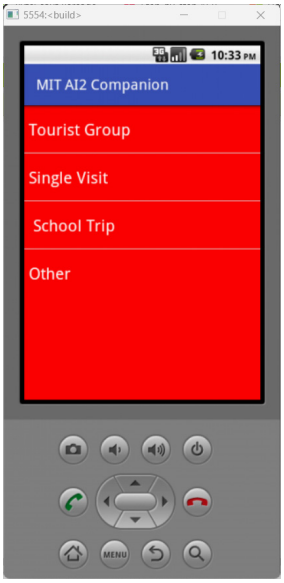
- ضبط عناصر المكوّن VisitList لتساوي قيمة المتغير ServicesList.



- ربط الكتل البرمجية لتصبح كالتالي:



9. حفظ التطبيق باسم KuwaitAttractions11 وعرضه من خلال المحاكي.





في الدرس القادم

سيتم استكمال تنفيذ النشاط، ويتضمّن ذلك:

- إضافة بيانات الزيارة باستخدام الزر BtnSave.
- عرض بيانات الزيارات المسجّلة باستخدام الزر BtnShow.



التطبيــق

ورقة عمل (11)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدع تطبيق HamadStore10.
2. حدد الشاشة PurchaseList.

أولاً: واجهة المصمم Design.

إضافة المكون ListPicker إلى الشاشة بالمكان المناسب ثم:

- تغييراسمه Rename إلى : Products.
- تغيير خصائصه التالية:

المطلوب	الخاصية
Blue	BackgroundColor
☑	FontBold
18	FontSize
Products	Text
Center:1	TextAlignment
White	TextColor
Choose the product	Title



ثانياً: واجهة الكتل البرمجية Blocks:

الانتقال إلى شاشة PurchaseList، ثم أضف الكتل البرمجية اللازمة لتنفيذ التالي:

- تسجيل أسماء المشترين.
- اختيار اسم المنتج من قائمة المنتجات.

3. أنشئ المتغير العام ProductsList: الذي يمثل قائمة المنتجات المتوفرة في المتجر: (T-shirt - Cap - Cup) .

4. أنشئ المتغير العام PurchaseDetails: ثم اربطه بقائمة فارغة لتخزين تفاصيل عملية الشراء.

5. أنشئ المتغير العام PurchasesList: ثم اربطه بقائمة فارغة لتخزين بيانات عملية الشراء.

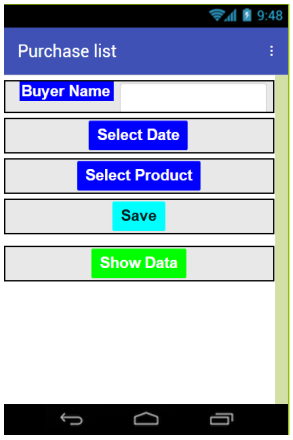
6. أنشئ المتغير العام BuyersList: لتخزين بيانات المُشترين.

7. أضف الكتل البرمجية اللازمة لتنفيذ التالي:

- عند فتح شاشة PurchasesList يتم تحميل جميع المنتجات المتوفرة في المتجر إلى المكون Products لتكون جاهزة للاختيار من قبل المستخدم.

8. حفظ التطبيق باسم HamadStore11 وعرضه من خلال المحاكي.







في وقت فراغك

صمّم تطبيقاً يحتوي على شاشة باسم FavoritList لإنشاء قائمة بالأماكن أو المنتجات المفضّلة لديك، وذلك باستخدام:

- القوائم Lists لإنشاء وتجميع البيانات.
- المكوّن ListPicker لعرض العناصر واختيار أحدها.
- كتلة join لدمج النصوص وعرض رسائل أو بيانات العناصر.



عبر عن رأيـك



أتعرف على مفهوم القوائم Lists كهيكل بيانات.	☐	☐	☐
أنشئ قائمة بيانات باستخدام الكتلة البرمجية make a list.	☐	☐	☐
أضيف عدد العناصر في القائمة حسب المطلوب باستخدام العلامة Mutator.	☐	☐	☐
أضيف عناصر جديدة للقائمة باستخدام الكتلة add items to list.	☐	☐	☐
أضيف المكوّن ListPicker وأغير خصائصه المرئية.	☐	☐	☐
أستخدم الفهرس Index للوصول إلى عناصر القوائم في App Inventor.	☐	☐	☐
أختبر تفاعل القوائم عبر المحاكي للتأكد من صحة العمل.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

تابع / التعامل مع القوائم - الفهرس

نواتج التعلم

- إضافة عناصر جديدة إلى القوائم باستخدام add item to list.
- إنشاء سجل باستخدام join لدمج البيانات.
- استخدام خاصية Selection لاسترجاع اختيار المستخدم.
- التعرف على شريط البحث ShowFilterBar داخل ListPicker.
- التمييز بين الحدثين BeforePicking و AfterPicking في إدارة القوائم.
- استخدام Length of list لعرض البيانات في LabShow.
- برمجة زر للتحقق من اكتمال البيانات قبل التسجيل.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

تحميل العناصر في ListPicker

استعرضنا في الدرس السابق طريقتين لتحميل العناصر في المكون ListPicker، وفي هذا الدرس سنستكمل الطريقة الثالثة التي تتعلق بكيفية التعامل مع اختيار المستخدم.

خاصية Selection

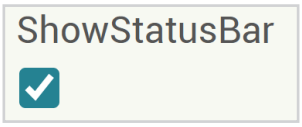
بعد اختيار المستخدم عنصرًا من القائمة داخل ListPicker، يتم تلقائياً تخزين القيمة التي تم اختيارها في خاصية تُسمّى Selection، وتُعد هذه الخاصية مهمة لأنها تُمكّن التطبيق من معرفة العنصر الذي اختاره المستخدم، مما يسمح باستخدامه في منطق التطبيق مثل:

- عرض تفاصيل العنصر المختار.
- أو استخدامه في عملية أخرى داخل البرنامج.

الخصائص الرئيسية ListPicker

يمتلك مكوّن ListPicker عددًا من الخصائص المهمة والتي تُساعد في التحكم في طريقة عرض القائمة وسلوكها عند الاستخدام، مثل:

- تفعيل خاصية شريط البحث (ShowFilterBar):



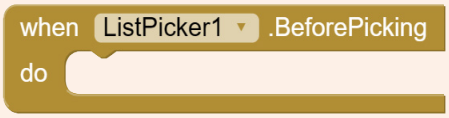
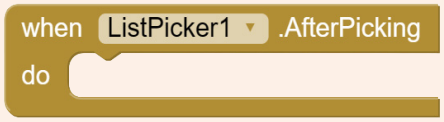
عند تفعيل هذه الخاصية، يظهر في أعلى القائمة شريط بحث يتيح للمستخدم المميزات التالية:



- البحث والتصفية: تصبح القائمة قابلة للبحث الفوري عن العناصر.
- تسهيل الوصول: يسهُل على المستخدم العثور على العنصر المطلوب بمجرد كتابة اسمه أو جزء منه.
- هذه الخاصية ضرورية جداً، خاصة عند التعامل مع القوائم الطويلة التي يصعب تصفحها يدوياً.

الأحداث الرئيسية ListPicker

يمتلك مكوّن ListPicker حدثين أساسيين يساعدان في التحكم بتدفق التطبيق قبل عرض القائمة وبعد اختيار المستخدم لعُنصر منها، وهما:

الحدث	الوصف	الاستخدام النموذجي
	يتم تشغيله قبل عرض شاشة القائمة المنبثقة للمستخدم.	يُستخدم لـ: ■ تعيين أو تحديث عناصر القائمة ديناميكيًا (باستخدام خاصية Elements). ■ تعيين عنوان القائمة (Title).
	يتم تشغيله بعد أن يختار المستخدم عنصراً من القائمة.	يُستخدم للحصول على: ■ قيمة الاختيار (Selection). ■ رقم فهرسة (SelectionIndex) وتنفيذ الإجراءات اللازمة بناءً على اختيار المستخدم.

■ ملخّص مبسّط:

- BeforePicking: تجهيز القائمة قبل عرضها.
- AfterPicking: معالجة نتيجة اختيار المستخدم.



النشـــاط

مهمة النشاط:

استكمال تطبيق KuwaitAttractions11 من خلال برمجة الزر BtnSave لإدخال وتخزين بيانات الزيارة، وبرمجة الزر BtnSwoh لعرض سجلات المواعيد المسجّلة، ثم اختبار عمل التطبيق بعد إضافة البيانات

تنفيذ النشاط:

1. تشغيل برنامج App Inventor، واستدعاء التطبيق KuwaitAttractions11 من مجلد Activity.
2. الانتقال إلى واجهة الكتل البرمجية ومن ثم اختيار شاشة VisitorRecord.

- **المرحلة الرابعة:** إضافة بيانات الزيارة من خلال الزر BtnSave.

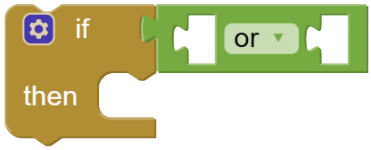
أولاً: برمجة زر الحفظ BtnSave (التحقق من البيانات والتسجيل).

- سحب كتلة الحدث للزر BtnSave



- التحقق من صحة البيانات باستخدام كتلة if...then...else مع أدوات المنطق or:
  حقل الاسم TextVisitor ليس فارغاً.
- VisitList اختيار نوع الزيارة.
- إذا كانت أي من هذه الحقول خالية، تظهر رسالة داخل المكوّن LabMsg:
  «You have to complete data»

- سحب كتلة الشرط if ومن ثم سحب الكتلة البرمجية من Logic وجمعها معا كما يلي :



- التحقق من ما إذا كان الحقل الأول فارغاً أو الثاني فارغاً، فسيتحقق الشرط:
  - مربع النص TextVisitor: التحقق مما إذا كانت خاصية Text تساوي نصاً فارغاً.



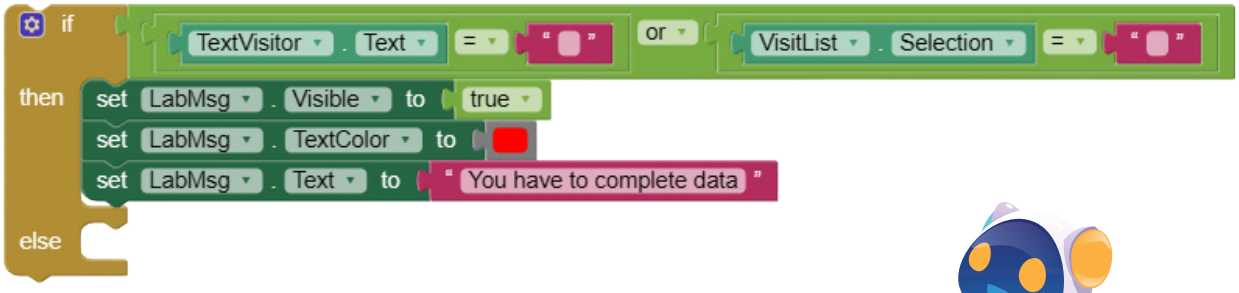
  - قائمة الاختيار VisitList : استخدام خاصية Selection للتأكد مما إذا كان المستخدم قد اختار عنصراً أم لا.



- تُجمع الكتل البرمجية كما يلي:



- تكون جملة if كالتالي:





ثانياً: إنشاء سجل الحجز .

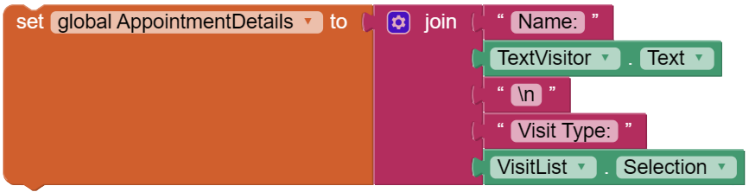
- تسجيل بيانات الزيارة، من خلال تجميع قيمة البيانات الأساسية التي أدخلها المستخدم وهي:

بيانات الزيارة	اسم الزائر	نوع الزيارة
تحديد قيمتها من	Textvisitor.Text	VisitList .Selection

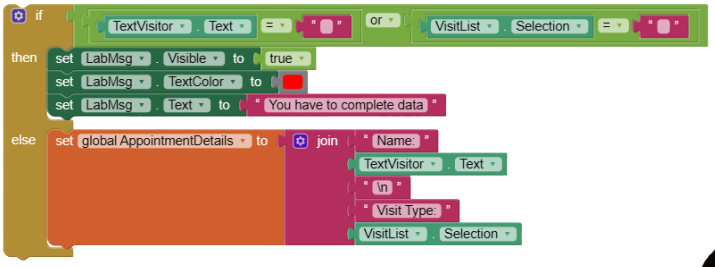
- تجميع البيانات من خلال استخدام الكتلة البرمجية join ، ويتم إضافة الكتلة البرمجية Text كعنوان للبيانات.

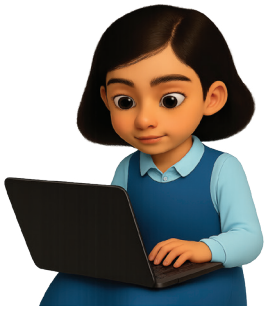


- إضافة النص الناتج من عملية التجميع داخل المتغير الذي يحتوي على تفاصيل بتفاصيل الزيارة AppointmentDetails.



- استكمال الكتلة البرمجية لجملة الشرط if (إضافة else من خلال Blue Gear Sign):



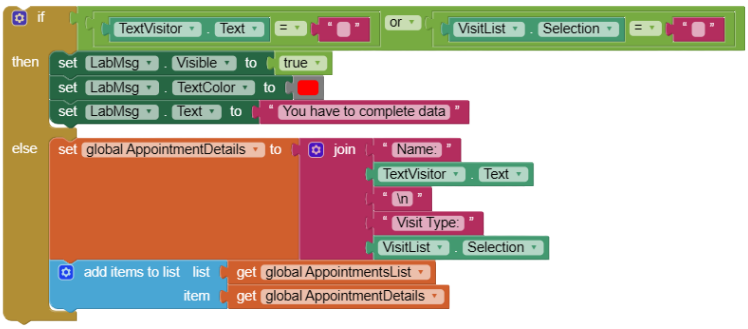


ثالثاً: الإضافة إلى القوائم Lists.

- إضافة بيانات الزيارة AppointmentDetails إلى قائمة المواعيد AppointmentsList.



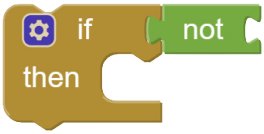
- تكون جملة if كالتالي:



رابعاً: إضافة الاسم إلى قائمة الزوار.

- التحقق من عدم تكرار الاسم من قبل وتكراره في القائمة:

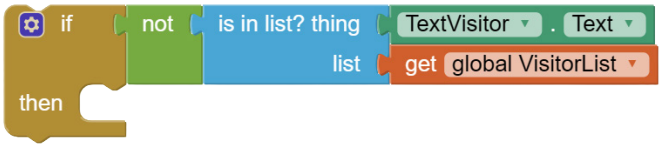
توظيف جملة if not من خلال إضافة الكتلة البرمجية not من تبويب Logic.



- إضافة الكتلة البرمجية is in list للتأكد من عدم وجود الاسم مسبقاً في القائمة list.



- عند التحقق من عدم تكرار الاسم يتم إضافته إلى قائمة الزوار.

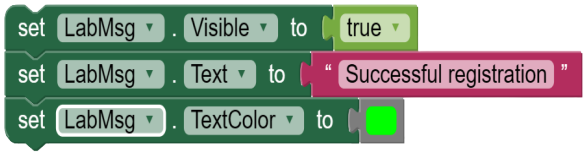


- إضافة عنصر جديد إلى القائمة باستخدام الكتلة add items to list ومن ثم تجميع الكتل كالتالي:

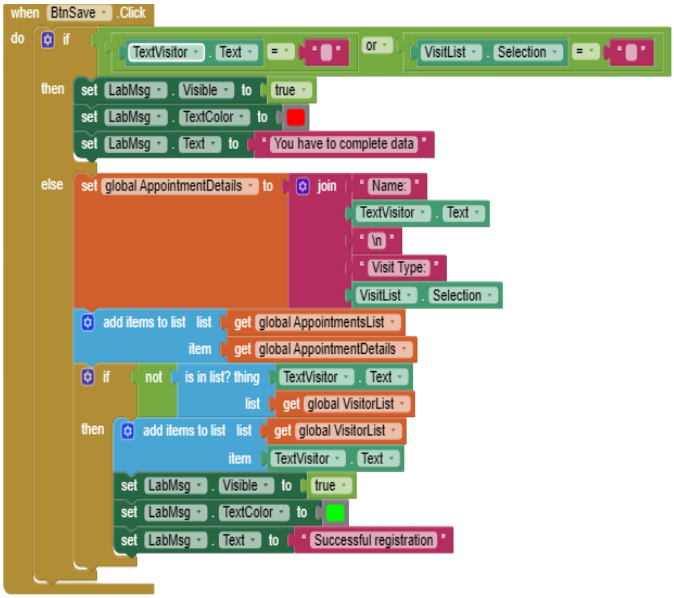


خامساً: إظهار رسالة نجاح للمستخدم.

- تأكيد عملية تسجيل الزيارة من خلال:
  - ظهور عبارة « Successful registration » داخل LabMsg.
  - تغير لون الرسالة إلى اللون الأخضر.



- الشكل النهائي للكتلة البرمجية للزر BtnSave.



- **المرحلة الخامسة:** عرض بيانات الزيارة من خلال الزر BtnShow.

سادساً: عرض بيانات الزيارة.

- سحب كتلة الحدث عند الضغط على الزر Btnshow:



- استخدام شرط if للتحقق من طول القائمة (قائمة المواعيد):



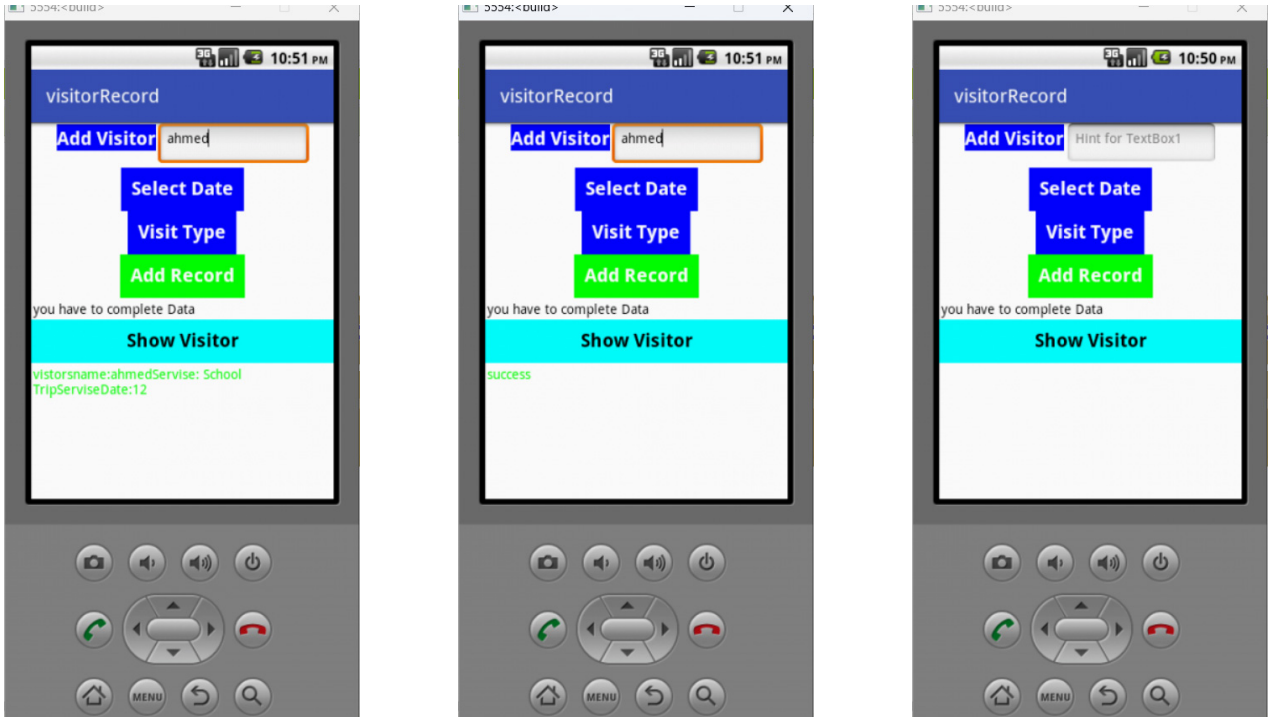
- عرض الزوار المسجّلين:
  - إذا كانت القائمة فارغة (Length of list=0) يظهر النص «No visitor» في LabShow.
  - أما إذا كان هناك زائرين تعرض البيانات في Labshow.



لاحظ

يمكن تغيير المتغير AppointmentDetail بالمتغير VisitorList لعرض الاسم فقط من بيانات الزوار.

3. حفظ التطبيق باسم KuwaitAttractions12 وعرض التطبيق من خلال المحاكي.





التطبيــق

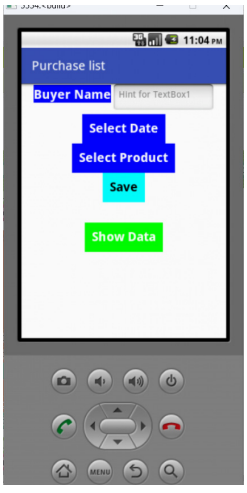
ورقة عمل (12)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدع تطبيق HamadStore11.
2. الانتقال إلى واجهة الكتل البرمجية (Blocks)، واختر الشاشة PurchaseList.
3. عند الضغط على زر BtnSave يجب تنفيذ المهام التالية:
   - االتحقق من إدخال اسم المشتري داخل المكوّن InputName.
   - التحقق من اختيار اسم المنتج داخل المكوّن (ListPicker) Products.
   - التأكد من اكتمال جميع المدخلات قبل عملية الحفظ.
   - حفظ بيانات عملية الشراء.
   - إظهار رسالة توضيح لنتيجة عملية التسجيل باستخدام (Label) SaveMsg:
     - إذا كانت البيانات ناقصة:

     رسالة تطلب إكمال الحقول « Please enter all purchase details» باللون الأحمر.
     - إذا كانت البيانات مكتملة:

     رسالة نجاح «Successful registration process» باللون الأخضر.









في وقت فراغك

صمم تطبيق FavoritesList

- إنشاء شاشة باسم FavoritesList.
- السماح للمستخدم بإدخال اسم المكان أو المنتج المفضل.
- إدخال سبب التفضيل أو تفاصيل مختصرة عنه.
- إضافة البيانات إلى قائمة المفضلات عند الضغط على زر الإضافة.
- ربط ListPicker بقائمة العناصر المضافة لعرضها عند الاختيار.
- عند اختيار عنصر من ListPicker يتم عرض الاسم مع التفاصيل باستخدام join داخل Label.



عبر عن رأيـك



أضيف عناصر جديدة إلى القوائم باستخدام add item to list	☐	☐	☐
أنشئ سجل باستخدام join لتجميع البيانات.	☐	☐	☐
أستخدم خاصية Selection لاسترجاع اختيار المستخدم.	☐	☐	☐
أتعرف على شريط البحث ShowFilterBar داخل ListPicker.	☐	☐	☐
أميز بين الحدثين BeforePicking و AfterPicking في إدارة القوائم.	☐	☐	☐
أستخدم Length of list لعرض البيانات في LabShow.	☐	☐	☐
أبرمج الزر للتحقق من اكتمال البيانات قبل التسجيل.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

الحلقات التكرارية Loops

نواتج التعلم

- شرح مفهوم الحلقات التكرارية وأهميتها في البرمجة.
- استخدام الكتلة البرمجية for Each.
- توظيف الكتلة for each number لإجراء تكرار محدد.
- توظيف الكتلة البرمجية while لتنفيذ تكرار يعتمد على شرط منطقي.
- تطبيق الحلقات التكرارية في مشاريع عملية.
- تمييز الفرق بين أنواع الحلقات التكرارية.
- تجنب الأخطاء الشائعة في استخدام الحلقات.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

الحلقات التكرارية Loops

هياكل برمجية تسمح بتنفيذ مجموعة من الأوامر البرمجية بشكل متكرر، وذلك بناءً على شرط معيّن.

ويتم إنشاء الحلقات التكرارية واستخدامها من خلال الكتل البرمجية الموجودة في تصنيف Control في App Inventor.

الحلقات التكرارية Loops

مميزات استخدام الحلقات التكرارية:

تقليل تكرار الكتل البرمجية اللازمة لتنفيذ مهمة واحدة عدة مرات



رفع كفاءة التطبيق من خلال تنفيذ المهام المتكررة بسرعة أكبر وأكثر تنظيمًا

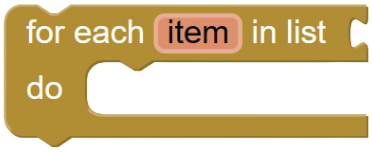
أنواع الحلقات التكرارية:

تنقسم الحلقات التكرارية من حيث طريقة عملها إلى ثلاثة أنواع:

1. for each item in list: تُستخدم لتنفيذ مجموعة من الأوامر لكل عنصر موجود داخل قائمة محددة.
2. for each number from ... to ... by ...: تكرار بعدد محدد من المرات.
3. while ... do: تكرار يستمر طالما الشرط صحيح.

أولاً: الحلقة التكرارية for each:

- الاستخدام:



تُستخدم لتنفيذ الأوامر مرة واحدة لكل عنصر داخل القائمة، ويكون عدد مرات التكرار معروفًا ومحددًا مسبقًا، وهي مثالية للعد من قيمة بداية إلى قيمة نهاية معينة، بزيادة أو نقصان محدد في كل خطوة.

- آلية العمل:

تُستخدم عند تنفيذ نفس الأوامر على كل عنصر داخل قائمة مثل: قائمة أرقام الهواتف، أسماء الطلاب، الدرجات .... وغيرها.

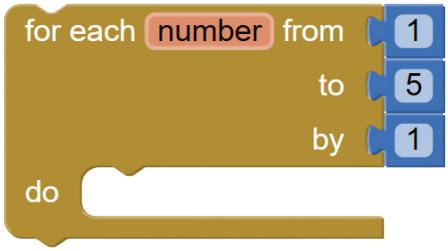
- عدد التكرارات مساويًا لعدد عناصر القائمة.
- يظهر داخل الحلقة متغيّر مؤقت مثل: (item) يمثّل العنصر الحالي من القائمة.
- ينتقل المتغير تلقائيًا من أول عنصر حتى آخر عنصر حتى يتم تنفيذ المهمة على جميع العناصر.

أمثلة:

- إرسال رسالة نصية لكل رقم في قائمة جهات اتصال.
- حساب مجموع درجات الطلاب الموجودة في قائمة واحدة.

ثانياً : الحلقة التكرارية for each number :

- **الاستخدام:**



تُستخدم عند الحاجة لتكرار الأوامر عددًا محددًا من المرات؛ حيث تمرّ على تسلسل من الأرقام يبدأ من قيمة From وينتهي عند قيمة To، مع زيادة أو نقصان ثابت يحدده المتغير by.

- **آلية العمل:**
  - ينتقل المتغير مثل: (number) خلال النطاق العددي خطوة بخطوة وفق قيمة by.
  - تمثّل قيمة هذا المتغير الرقم الحالي المستخدم في تنفيذ الأوامر داخل جزء do.
  - يمكن تغيير اسم المتغير (number) إلى أي اسم آخر يناسب المهمة، مثل: i أو count.

مثال:

- عرض جدول الضرب لعدد يختاره المتعلم (من 1 إلى 10).
- تكرار أمر تغيير لون عنصر في الشاشة 5 مرات متتالية.

ثالثاً : الحلقة التكرارية while do :

- **الاستخدام:**



تُستخدم عندما لا يكون عدد مرات التكرار معروفًا مسبقًا، ويعتمد استمرار التكرار على تحقق شرط منطقي مثل:

- طالما عدد المحاولات أقل من 3.
- طالما المستخدم لم يُدخل الإجابة الصحيحة.

- **آلية العمل:**
  - تبدأ الحلقة بفحص الشرط.
  - إذا كان الشرط صحيح (True): تُنفّذ الأوامر داخل الحلقة.
  - بعد التنفيذ تعود الحلقة لإعادة فحص الشرط مرة أخرى.
  - إذا أصبح الشرط غير صحيح (False): تتوقف الحلقة وتُكمِل الأوامر التالية لها.

هـذه الآليـة تجعـل الحلقـة مناسـبة للمواقـف التـي تعتمـد علـى تفاعـل المسـتخدم، أو الانتظـار حتـى يتحقـق شـرط معيّـن.

أمثلة:

يمكـن اسـتخدام حلقـة while لتكـرار عمليـة حتـى يصـل المتغيّـر إلـى قيمـة معينـة، مثـل:

- جمع الأعداد من 1 إلى N داخل متغيّر.
- الاستمـرار فـي طلـب إدخـال كلمـة السـر حتـى يكتب المستخدم القيمة الصحيحة.

الأخطاء البرمجية الشائعة في الحلقات التكرارية

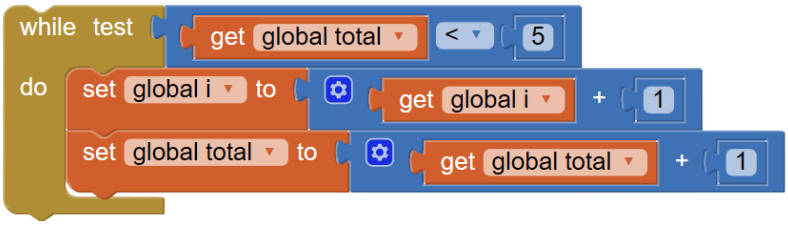
■ نسيان زيادة الفهرس Counter داخل While

هـو السـبب الأكثـر انتشـارًا لحـدوث الحلقـات اللانهائيـة Infinite Loop، حيـث يظـل الشـرط محققـاً دائمـا لأن قيمـة الفهـرس داخـل الحلقـة لا تتغيـر.

الخطأ:



تصحيح الخطأ:



■ استخدام for each مع متغيّر ليس قائمة

هـذه الحلقـة مصممـة للمـرور علـى عناصـر قائمـة (مثـل قائمـة مـدن قائمـة منتجـات -.... الـخ)، واسـتخدامها مـع رقـم أو نـص بـدل القائمـة يـؤدي إلـى خطـأ فـي التشـغيل، أو عـدم تنفيـذ الحلقـة إطلاقًـا، لأن الحلقـة لا تجـد عناصـر للمـرور عليهـا.

كتابة شرط خاطئ في While

قـــد يمنع الحلقة مـــن العمل نهائيًا إذا كان الشـــرط خاطئًا من البداية، أو يســـبّب خطأ Off-by-one (تكـــرار ناقص أو زائد).

اختيار حلقة تكرارية غير مناسبة

- اســـتخدام for range عندما تكون المهمة تعتمد على شـــرط خارجـــي أو متغير (مثل انتظار إدخال المســـتخدم) يجعل الكتل البرمجيـــة غير مرنة.
- أو استخدام while عندما يكون التكرار محددًا بعدد يؤدي إلى تعقيد غير ضروري.

كيفية تجنب هذه الأخطاء:

- **مبدأ التحديث المستمر :** الحـرص علـى أن يكـون أول سطـر يُكتب داخل while هو تحديث الفهرس أو المتغير المستخدم في الشرط.
- **التحقق من البيانات:** التأكـد دائمـاً أن المتغيـر الذي سيتـم استخدامـه في for each هو بالفعل قائمة List .
- **المراجعة المنطقية:** مراجعـة الشـرط والتأكـد مـن أنـه يؤدي إلى توقف الحلقة في الوقت المناسب.



نشـــاط 1

مهمة النشاط:

اســتكمال نشــاط تطبيــق KuwaitAttractions12 مــن خلال إدراج مجموعــة مــن معالــم دولــة الكويــت فــي قائمــة List، وتطبيــق نــوع مناســب مــن الحلقــات التكراريــة للمــرور علــى العناصــر وعــرض (كل معلــم مــع تاريــخ الافتتــاح فــي ســطر منفصــل) .

تنفيذ النشاط :

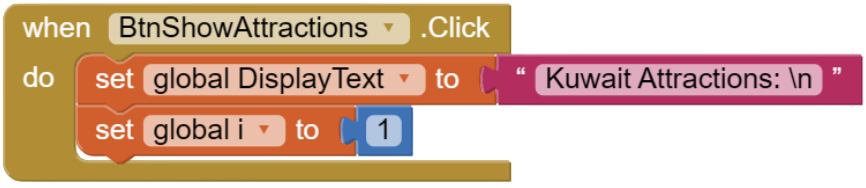
1. تشغيل برنامج App Inventor واستدعاء التطبيق KuwaitAttractions12 من مجلد الأنشطة Activity.
2. الانتقال إلى واجهة الكتل البرمجية Blocks، وتحديد شاشة historicalknowledge.

- المتغيرات العامة الموجودة في التطبيق:

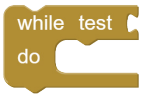
المتغير	ما يمثله
Attractions	قائمة المعالم السياحية.
CurrentAttraction	تخزين بيانات المعالم السياحية (الاسم - تاريخ الافتتاح).
i	فهرس التكرار للانتقال بين عناصر القائمة.
LineText	بناء سطر نصي لمعلومة واحدة عن المعلم السياحي: رقم التسلسل - المَعلم - السنة.
DisplayText	تجميع السطور النصية التي تم عرضها داخل المكون LblAttractions .
CurrentYear	سنة انشاء المعلم السياحي الحالي.
Years	قائمة سنوات الافتتاح لكل معلم .

3. تحديد الزر BtnShowAttractions وإدراج الكتل البرمجية لعرض المعالم السياحية بدولة الكويت داخل المكون LblAttractions:

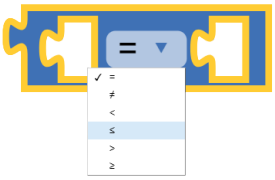
- استكمال كتلة الحدث عند الضغط على الزر:
  - إعداد عنوان القائمة قبل البدء في عرض العناصر في displayText.
  - تعيين قيمة الفهرس i إلى 1 للبدء من العنصر الأول في القوائم.



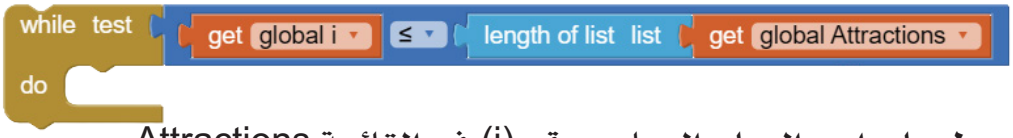
- إضافة الحلقة التكرارية While test.



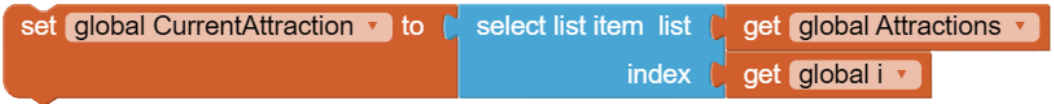
- إضافة الكتلة البرمجية من قسم Math لمقارنة قيمة المتغير i و طول القائمة Attractions.



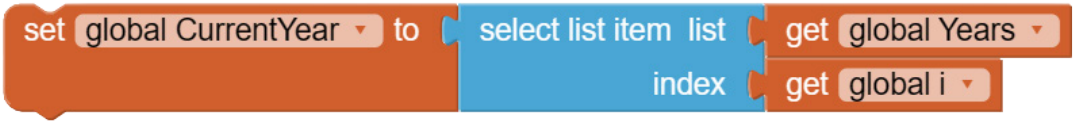
- تجميع الكتل.



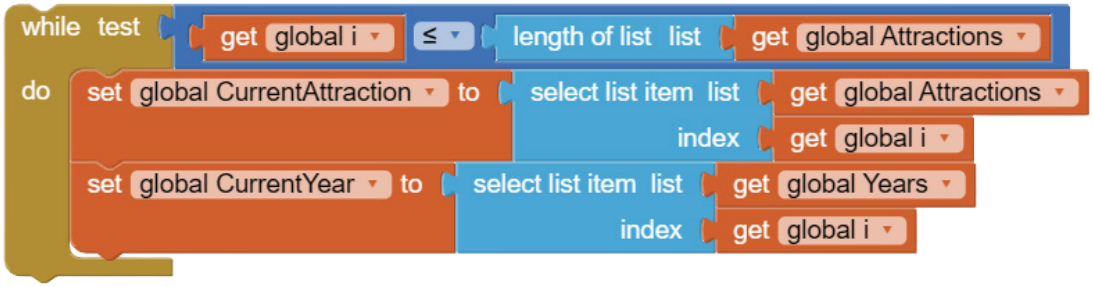
- الحصول على اسم المعلم السياحي رقم (i) في القائمة Attractions.



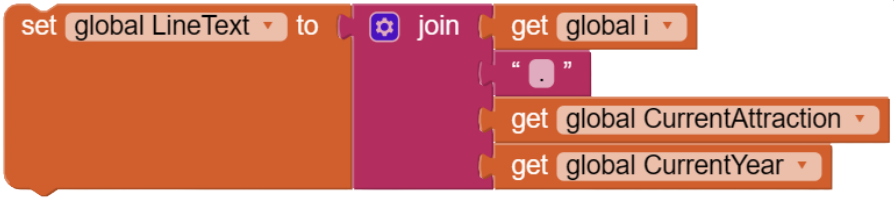
- الحصول على اسم سنة افتتاح المقابلة للعنصر رقم (i) في القائمة Years.



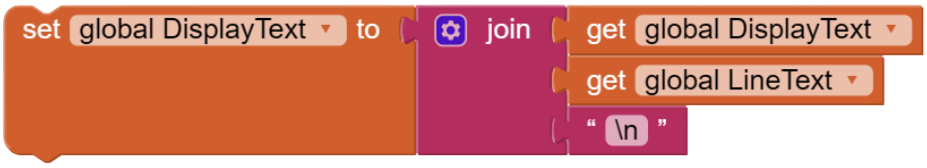
- استخراج العنصر رقم (i) من قائمتين مختلفتين Attractions ثم Years، وتخزينه مؤقتا في المتغيرين CurrentAttraction و Years.



- بناء سطر نصي يمثل معلومة واحدة عن المعلم السياحي: رقم التسلسل - المَعلم - السنة.



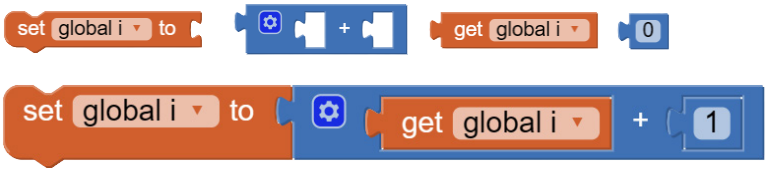
- إلحاق السطر النصي بالنصّ الكامل وإضافة سطر جديد.



لاحظ

عند إضافة \n يتم الانتقال إلى سطر جديد.

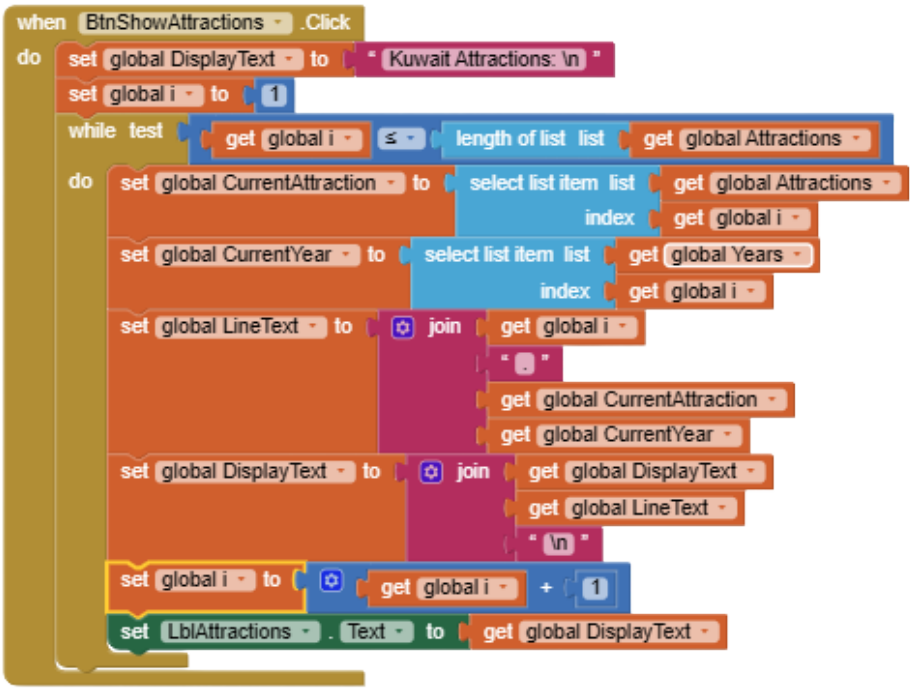
- زيادة قيمة الفهرس i بمقدار 1 للانتقال إلى العنصر التالي في القائمة :
  - سحب الكتل التالية ومن ثم تجميعها :



- عرض محتوى المتغير DisplayText داخل المكوّن LblAttractions في الشاشة.



- تجميع الكتل البرمجية BtnShowAttractions :



4. حفظ التطبيق باسم KuwaitAttractions13 وعرض التطبيق من خلال المحاكي:





التطبيـــق

ورقة عمل (13)

من خلال دراستك لبرنامج / منصة MIT App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore12 من مجلد أوراق العمل Workpaper9_2.
2. انتقل إلى واجهة الكتل البرمجية Blocks.
3. اختر شاشة StoreData.
4. أنشئ قائمة بالمنتجات تحتوي على البيانات التالية:

السعر	الرقم المسلسل	اسم المنتج

وذلك كالتالي:

5. عند الضغط على زر BtnAllProd:
   - يتم عرض قائمة المنتجات في المكون LblproductsList.
   - مستعينا بالجدول التالي والخاص بالمتغيرات:

المتغير	ما يمثلـــــه
CurrentProducts	قائمة بيانات المنتجات (الاسم - السعر).
LineText	تجميع سطر نصي لبيانات كل منتج.
i	فهرس التكرار للانتقال بين عناصر القائمة.
DisplayText	تجميع السطور النصية التي تم عرضها داخل المكون LblproductsList.
CurrentPrice	سعر المنتج الحالي.
Products	قائمة بأسماء المنتجات.
Price	قائمة أسعار المنتجات.

6. احفظ التعديلات باسم HamadStore13 ثم أعرض التطبيق من خلال شاشة المحاكي.





في وقت فراغك



- استرشد بمواقع الذكاء الاصطناعي التوليدي ثم أدرج معالم إضافية من معالم دولة الكويت وطبق ما تعلمته في هذا الدرس من حيث الاستعلام عن تاريخ إنشاء المعلم.



عبر عن رأيــك



أشرح مفهوم الحلقات التكرارية وأهميتها في البرمجة.	☐	☐	☐
أستخدم كتلة التكرار for Each.	☐	☐	☐
أوظف الكتلة البرمجية For range للتكرار بعدد محدد.	☐	☐	☐
أوظف الكتلة البرمجية while لتنفيذ عمليات تكرار شرطية .	☐	☐	☐
أطبق حلقات التكرار في مشاريع عملية.	☐	☐	☐
أميز الفرق بين الحلقات التكرارية المختلفة.	☐	☐	☐
أتجنب الأخطاء الشائعة في استخدام الحلقات.	☐	☐	☐



ملاحظات المعلم

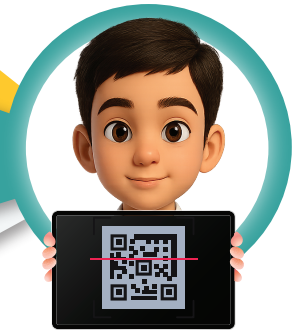
اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

الإجراءات Procedures

نواتج التعلم

- شرح مفهوم الإجراء Procedure.
- إنشاء إجراءات مخصصة مع أو بدون معطيات Parameters.
- توضيح فكرة المعطيات Arguments داخل الإجراء وكيفية استخدامها.
- تصميم الإجراء SearchAttraction للبحث عن معلم سياحي.
- استخدام الكتل trim - downcase لمعالجة نص الإدخال.
- تنظيم الكتل البرمجية باستخدام الإجراءات.
- توظيف الإجراءات لتنفيذ مهام متكررة.
- تطبيق مبدأ إعادة الاستخدام في البرمجة.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

الإجراءات Procedures

طريقة تنظيم الأوامر البرمجية في «وحدات» يمكن إعادة استخدامها أكثر من مرة داخل التطبيق دون تكرار الكتل البرمجية.

تعريف الإجراء Procedure:

مجموعة مرتبة من التعليمات البرمجية تنفّذ مهمة محددة دون إرجاع قيمة ، مثل: البحث عن معلم سياحي أو عرض معلومات منتج، ويمكن استدعاء الإجراء بالاسم في أي وقت داخل التطبيق.

الإجراءات Procedures

أهداف استخدام الإجراءات:

ترتيب الكتل البرمجية.	تقليل التكرار.	إمكانية تعديل التطبيق بسهولة.

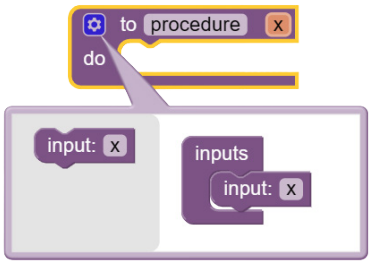
- في برنامج App Inventor توجد كتل الإجراءات داخل تصنيف Procedures في لوحة الكتل البرمجية Blocks.
- ويمكن أن تحتوي على معطيات Arguments لتغيير سلوكها حسب الحاجة.

إنشاء الإجراء في App Inventor

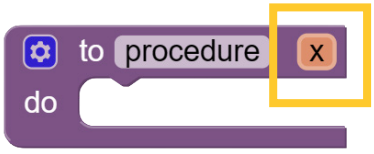


- من لوحة الكتل البرمجية Blocks.
- الانتقال إلى تصنيف Procedures.
- سحب كتلة الإجراء to procedure إلى منطقة العمل.
- تسمية الإجراء (اسم مناسب ومعبّر وله دلاله).
- إضافة المدخلات (اختياري).
  - اسم المعلَم السياحي.
  - رقم المنتج.
  - السعر.
  - أو أي قيمة يريدها المبرمج.

تُسمّى هذه البيانات مدخلات Arguments، ويتم إضافتها بالضغط على علامة Blue Gear Button.



أصبح للإجراء مدخل يمكن استخدامه داخل الكتلة بالشكل التالي:



- يتم سحب الكتل البرمجية التي تنفّذ المهمة ووضعها داخل الإجراء.

معالجة النصوص داخل التطبيق

من أجل الوصول إلى معالجة صحيحة للنصوص، ولتحقيق مقارنة دقيقة بين الكلمات أو المدخلات النصية داخل التطبيق، نستخدم كتلاً تساعد لتحسين النص قبل مقارنته من الكتل البرمجية Text:

كتلة trim:



- تُستخدم لإزالة المسافات الزائدة في بداية النص ونهايته.
- تمنع الأخطاء الناتجة عن وجود فراغات غير مقصودة عند إدخال البيانات.

كتلة downcase:



- تُحوّل جميع الأحرف في النص إلى أحرف صغيرة.
- تضمن توحيد صيغة النص عند المقارنة، مما يزيد من دقة المطابقة بين المدخلات.

النتيجة المتوقعة:

يسهم استخدام كل من trim و downcase في رفع دقة المقارنة النصية داخل التطبيق، والحد من الأخطاء الناتجة عن اختلاف حالة الأحرف أو وجود مسافات إضافية.



نشاط 1

مهمة النشاط:

استكمال نشاط الدرس السابق تطبيق KuwaitAttractions13 من خلال:

البحث عن اسم المعلَم السياحي الذي يدخله المستخدم ، ثم عرض بيانات هذا المعلم (الاسم وتاريخ الافتتاح) وذلك باستخدام الإجراءات Procedures.

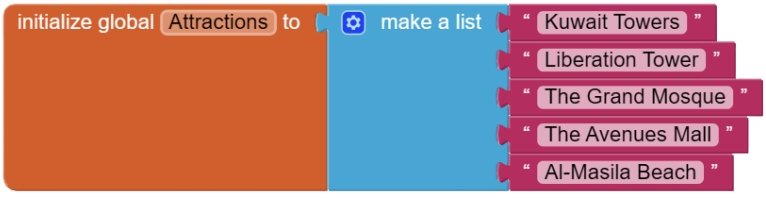
تنفيذ النشاط :

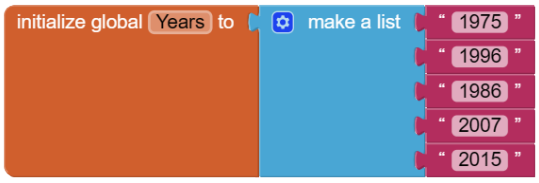
1. تشغيل برنامج App Inventor واستدعاء التطبيق historicalknowledge من مجلد الأنشطة Activity.
2. الانتقال إلى واجهة الكتل البرمجية Blocks، وتحديد شاشة TouristAttractions.

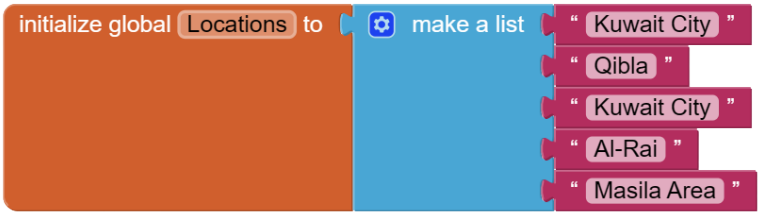
- المتغيرات الموجودة في التطبيق:

المتغير	ما يمثله
foundIndex	يحدد موقع المعلَم السياحي الذي تم العثور عليه داخل القائمة. initialize global foundIndex to
i	فهرس يستخدم للتنقّل بين عناصر القائمة أثناء البحث. initialize global i to
found	متغير منطقي يوضح هل تم العثور على المعلَم السياحي؟ initialize global found to

وبالاستعانة بالقوائم المستخدمة في الشاشة:









■ **المرحلة الأولى : بناء إجراء البحث searchAttraction**

3. إنشاء وتجهيز الإجراء The Procedure.

■ سحب كتلة to procedure من قائمة Procedures وتسميتها باسم SearchAttraction:



4. ضبط قيم المتغيرات قبل بدء بعملية البحث وترتيبها داخل الإجراء كالتالي:

■ ضبط القيمة الابتدائية للمتغير foundIndex = 0

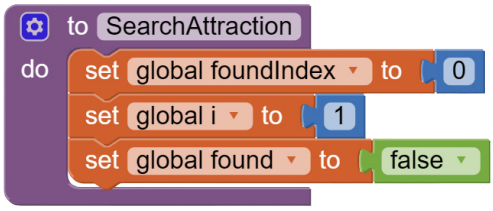
(للدلالة على أنه: لم يتم العثور على المعلم بعد)

■ ضبط القيمة الابتدائية للمتغير i = 1

(تعني: أن البحث يبدأ من أوّل عنصر)

■ ضبط القيمة الابتدائية للمتغير found = false

(تعني: حتى الآن لم يتم العثورعلى المَعلم)



5. استخدام حلقة التكرار while test مع وضع شرطين:

■ أن يكون الفهرس (العداد) i أصغر من أو يساوي طول القائمة .



■ أن تكون قيمة found لا تزال false.



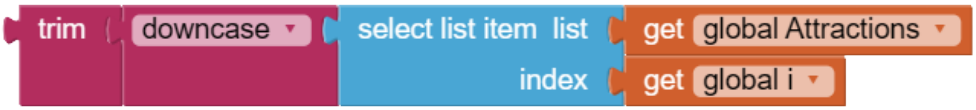
■ تجميع الشرطين داخل كتلة المقارنة and، ومن ثم إضافتها داخل الحلقة التكرارية.



6. معالجة النص المدخل من قبل المستخدم قبل البدء بالمقارنة لضمان تطابق النتائج ، باستخدام كتل النصوص التالية:

- استخدام الكتلتين trim و downcase.

(تحويل اسم المعلم من القائمة إلى أحرف صغيرة وإزالة المسافات)



- استخدام الكتلة downcase.

(تحويل اسم المعلم الذي أدخله المستخدم إلى أحرف صغيرة)



- مقارنة الاسم داخل القائمة بالاسم الذي كتبه المستخدم ومن ثم ربط كتلة المقارنة مع الشرط if.



7. عند العثور على اسم المَعلم: (أي تحقق التطابق - يتم إيقاف البحث)

- تحديث قيمة foundIndex ليأخذ قيمة العداد i الحالي.



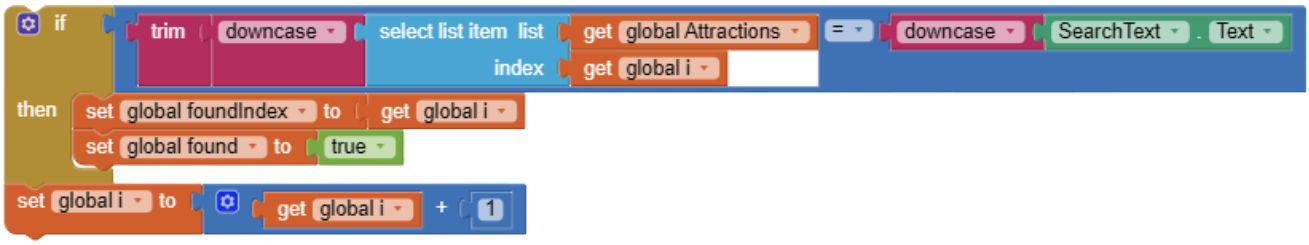
- تغيّر قيمة found إلى true لإيقاف البحث.



- تحديث العداد بإضافة 1 إلى قيمة i في نهاية الحلقة للانتقال للعنصر التالي.



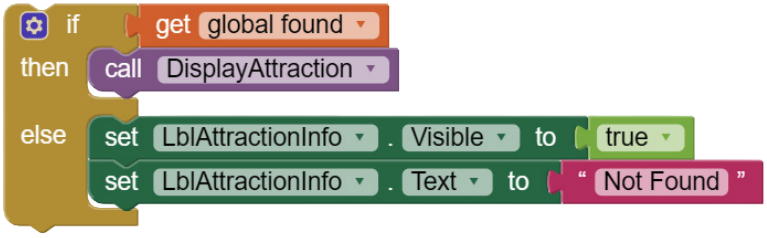
- تجميع الكتل داخل الشرط if كالتالي.



- **المرحلة الثانية : عرض النتائج .**

8. بعد انتهاء حلقة البحث (داخل نفس الإجراء)، يتم إضافة كتلة الشرط if لعرض النتيجة للمستخدم:

- إذا كان found يساوي true : يتم استدعاء الإجراء DisplayAttraction (المجهز مسبقاً) لعرض البيانات.
- وإلا (إذا كان found يساوي false) : يتم عرض النص Not Found داخل LblAttractionInfo .



9. تجميع الكتل البرمجية كاملة للإجراء searchAttraction.





- **المرحلة الثالثة :** ربط الزر BtnSearch بالبرنامج.

10. استدعاء الإجراء SearchAttraction فقط وظهور اسم المعلم السياحي في LblAttractionInfo :

- سحب كتلة الحدث للزر.
- إضافة كتلة واحدة داخلها لاستدعاء الإجراء SearchAttraction.

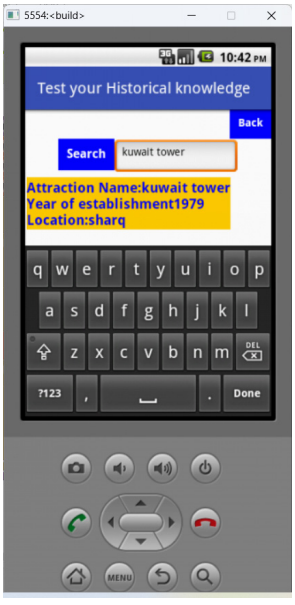


تذكر

استخدام الإجراء DisplayAttraction وهو موجود مسبقًا ويقوم بالتالي:

- جمع النصوص (اسم المَعلم، سنة تأسيسه، الموقع) معاً: باستخدام join.
- إضافة سطر جديد : باستخدام \n.
- عرض البيانات داخل LblAttractionInfo.

11. حفظ التطبيق باسم KuwaitAttractions14 وعرضه من خلال المحاكي.





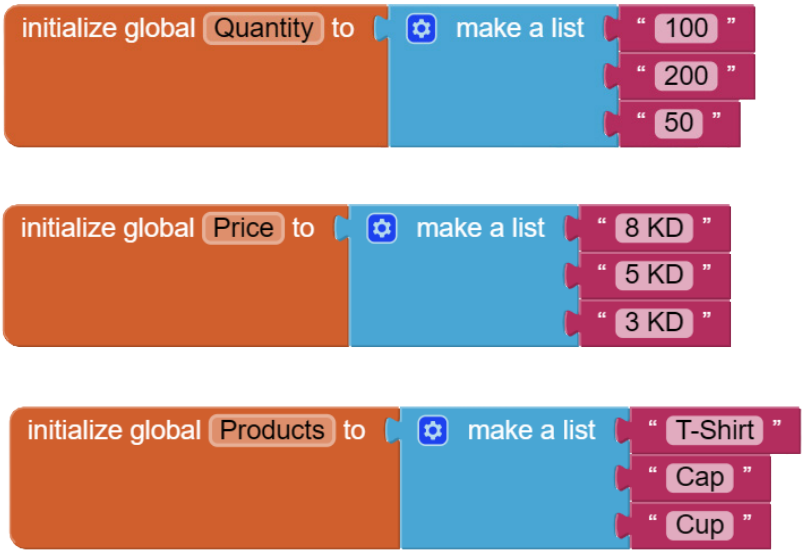
التطبيـــق

ورقة عمل (14)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

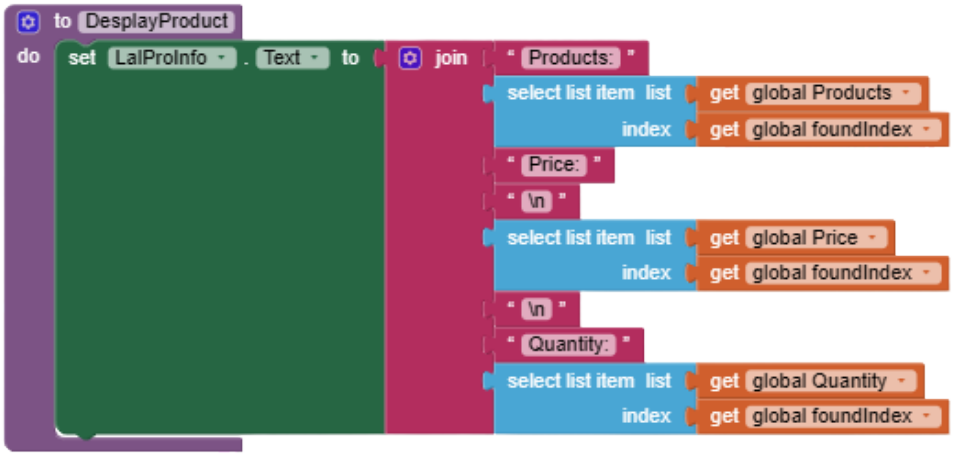
1. استدعِ التطبيق HamadStore13 من مجلد أوراق العمل Workpaper9_2.
2. انتقل إلى واجهة الكتل البرمجية Blocks، ثم اختر شاشة StoreData.
3. أنشئ إجراء Procedures باسم SearchProducts للبحث عن المنتجات كالتالي:
   - كتابة اسم المنتج في TextPro.
   - الضغط على زر BtnSearch.
     - إذا كان المنتج متوفر يظهر في LblProInfo.
     - وإذا كان المنتج غير متوفر تظهر رسالة Not Found.

مستعينا بالكتل البرمجية الموجودة بالتطبيق والخاصة بالمتغيرات والقوائم كما بالشكل التالي:





ومسترشداً بالإجراء: Procedures displayProduct.



4. احفظ التعديلات باسم HamadStore14 ثم أعرض التطبيق من خلال شاشة المحاكي.



في وقت فراغك



- انشاء تطبيق يتضمن شاشة جديدة مستخدما ثلاثة إجراءات:
  - إجراء لعرض رسالة ترحيب منسّقة في Label معيّن.
  - إجراء لتنسيق أي نص يُعرض في التطبيق بإضافة سطر جديد.
  - إجراء لمسح محتويات مجموعة من حقول الإدخال مرة واحدة بدل مسح كل حقل بكتلة منفصلة.
  - ربط هذه الإجراءات بأزرار مختلفة (زر ترحيب، زر تنسيق، زر مسح).
  - اختبار كيفية استدعاء نفس الإجراءات في أكثر من مكان داخل التطبيق بدون تكرار الكتل البرمجية.



عبر عن رأيــك



أشرح مفهوم الإجراء Procedure.	☐	☐	☐
أنشئ إجراءات مخصصة مع أو بدون معطيات Parameters.	☐	☐	☐
أوضح فكرة المعطيات Arguments داخل الإجراء وكيفية استخدامها.	☐	☐	☐
أصمم الإجراء searchAttraction للبحث عن معلم سياحي.	☐	☐	☐
أستخدم الكتل trim - downcase لمعالجة نص الإدخال.	☐	☐	☐
أنظم الكتل البرمجية باستخدام الإجراءات.	☐	☐	☐
أوظف الإجراءات لتنفيذ مهام متكررة.	☐	☐	☐
أميز بين الإجراءات و الدوال.	☐	☐	☐



ملاحظات المعلم

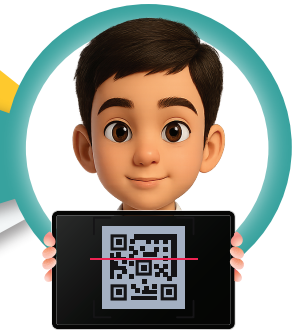
اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

الدوال Functions

نواتج التعلم

- شرح مفهوم الدوال وأهميتها في البرمجة.
- إنشاء دوال تُرجع قيماً بعد تنفيذ العمليات.
- استخدام الدوال في الحسابات والمعالجة المنطقية.
- التمييز بين الإجراءات Procedures والدوال Functions.
- دمج الدوال مع الإجراءات لبناء تطبيق فعّال.
- تطبيق الدوال في حل مسائل رياضية أو منطقية.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

الدالة Function

الدالـــة « أداة ذكيـــة » نمـــرر لهـــا بيانـــات مـــدخلات، فتعمـــل علـــى معالجتهـــا ثـــم تعيـــد لنـــا النتيجـــة مخرجـــات.

الدالـــة تشـــبه الآلـــة الحاســـبة تمامًـــا، تجعـــل الأوامـــر البرمجيـــة (الكتـــل البرمجيـــة) أقصـــر، أوضـــح، وأســـهل فـــي التعديـــل.

الفرق بين الإجراءات والدوال

جدول مقارنة بين الإجراءات والدوال		
القيمة	الإجراءات Procedures	الدوال Functions
القيمة	لا تُرجع قيمة	ترجع قيمة
الاستخدام	تنفيذ مهمة (عرض - تحديث - تغيير)	حساب أو تقييم قيمة
الكتلة البرمجية المستخدمة	to function + return	to procedure

لاحظ

- لا يوجد في App Inventor تصنيف مستقل باسم Functions.
- الدوال المدمجة Built-in Functions موجودة ضمن تصنيفات مثل: Math - Text - Logic.
- الدوال يتم إنشاؤها من قبل المستخدم تُبنى من تصنيف Procedures باستخدام كتلة return.





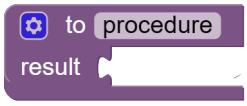
أمثلة على الدوال المدمجة

دالة Contains text من تصنيف Text

تستخدم مطابقة أجزاء من النص بدلاً من المطابقة الكاملة للنصوص حيث يمكن استخدام جزء من النص.

مثال: يكتب المستخدم (Tower) فقط أو أي جزء من النص، بدلاً من كتابة (Kuwait Towers) كاملة.

- المكونات:
  - text: النص الأصلي (كما هو مسجل في البرنامج).
  - piece: الجزء المراد البحث عنه (الاسم المدخل عن طريق المستخدم).





النشـــاط

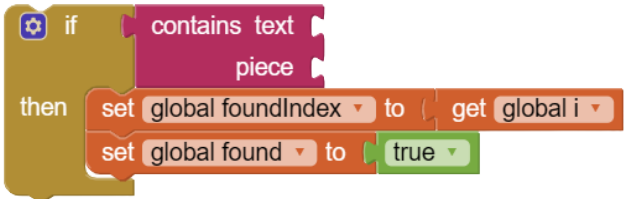
مهمة النشاط:

- استكمال نشاط الدرس السابق تطبيق KuwaitAttractions14.
- تطوير عملية البحث عن المَعلم السياحي ، بحيث يتمكن المستخدم من البحث باستخدام جزء من الاسم.

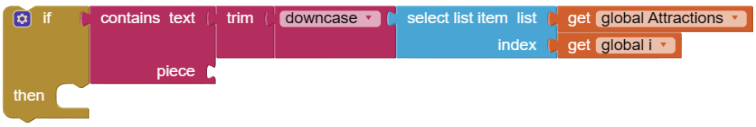
تنفيذ النشاط:

1. تشغيل برنامج App Inventor، واستدعاء التطبيق KuwaitAttractions14 من مجلد الأنشطة Activity.
2. الانتقال إلى واجهة الكتل البرمجية Blocks، وتحديد شاشة historicalknowledge.
3. تعديل الإجراء Procedures: SearchAttraction.

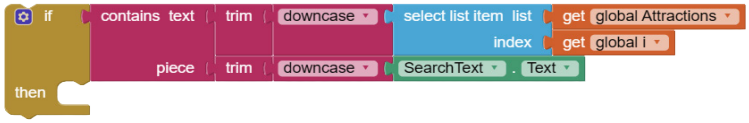
- تعديل شرط جملة if:
  - استبدال المقارنة (=) بالكتلة contains text.
    - النص الكامل text: عناصر قائمة أسماء المعالم.
    - ما يبحث عنها المستخدم piece: SearchText.



- التحقق من القائمة:
  - التحقق مما إذا كان جزء من الاسم الذي كتبه المستخدم موجوداً داخل القائمة Attractions المخزنة في التطبيق:
    - في جزء text: إضافة قائمة أسماء المعالم المخزنة في التطبيق.



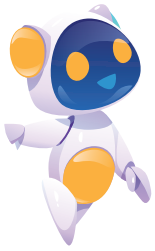
    - في جزء piece: إضافة ما سيتم كتابته من قبل المستخدم داخل مربع النص SearchText.



  - الشكل النهائي للكتل البرمجية الخاصة بالإجراء SearchAttraction: Procedures



4. حفظ التطبيق باسم KuwaitAttractions15 وعرض التطبيق من خلال المحاكي.







التطبيــق

ورقة عمل (15)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدع تطبيق HamadStore14.
2. الانتقال إلى واجهة الكتل البرمجية (Blocks)، واختر الشاشة StoreData.
3. طوّر الإجراء Procedures : SearchProduct ليتمكن المستخدم من البحث عن المنتج بجزء من الاسم بدلاً من الاسم كاملاً.
4. احفظ التعديلات باسم HamadStore15 ثم أعرض التطبيق من خلال شاشة المحاكي.







في وقت فراغك

استكمل ما تم تنفيذه في وقت فراغك السابق بحيث يتم توظيف الدوال لمسح محتويات مجموعة من حقول الإدخال مرة واحدة بدل مسح كل حقل بكتلة منفصلة.



عبر عن رأيـك



أشرح مفهوم الدوال وأهميتها في البرمجة.	☐	☐	☐
أنشئ دوال تُرجع قيماً بعد تنفيذ العمليات.	☐	☐	☐
أستخدم الدوال في الحسابات والمعالجة المنطقية.	☐	☐	☐
أميز الدوال عن الإجراءات بوضوح.	☐	☐	☐
أدمج الدوال مع الإجراءات لبناء تطبيقات فعّالة.	☐	☐	☐
أطبق الدوال في حل مسائل رياضية أو منطقية.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

مدخل إلى قواعد البيانات TinyDB

نواتج التعلم

- شرح مفهوم قاعدة البيانات المحلية.
- توضيح أهمية TinyDB في حفظ البيانات.
- تمييز الفرق بين الذاكرة المؤقتة (المتغيرات) والذاكرة الدائمة TinyDB.
- استخدام كتل StoreValue و GetValue لحفظ واسترجاع البيانات.
- تطبيق TinyDB لحفظ إعدادات أو معلومات مستخدم في تطبيقات واقعية.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

TinyDB

مكوّن غير مرئي يعمل كقاعدة بيانات محلية لتخزين البيانات بشكل دائم على جهاز المستخدم، بحيث تبقى البيانات موجودة حتى بعد إغلاق التطبيق أو إعادة تشغيل الجهاز.

مدخل إلى قواعد البيانات TinyDB

تُخزّن البيانات داخله في صورة أزواج : الوسم والقيمة (Value - Tag) حيث يعمل الوسم مثل اسم أو مفتاح نستخدمه لاحقاً لاسترجاع نفس البيانات:

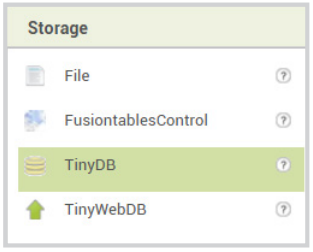
- الوسم (Tag): اسم مميز يعرّف مجموعة البيانات.
- القيمة (Value): البيانات نفسها المرتبطة بهذا الوسم.



مثال		
Tag	اسم المستخدم	درجة الطالب
Value	أحمد	95

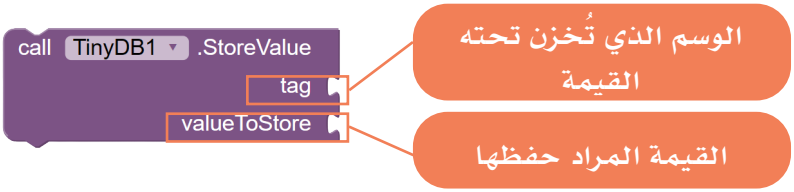
إضافة TinyDB في App Inventor

- من تصنيف Storage اختيار TinyDBحيث يظهر في أسفل المكونات (غير مرئي أثناء التشغيل).



حفظ البيانات

- تُستخدم الكتلة البرمجية StoreValue لحفظ البيانات داخل قاعدة البيانات.



هيكلة البيانات بنظام الوسوم Tags

تعتمد أداة TinyDB على تخزين عناصر البيانات كسلاسل نصية مرتبطة بوسوم Tags محددة، مما يسهل عملية تحديد واسترداد المعلومات بدقة.

وحدة مخزن البيانات للتطبيق

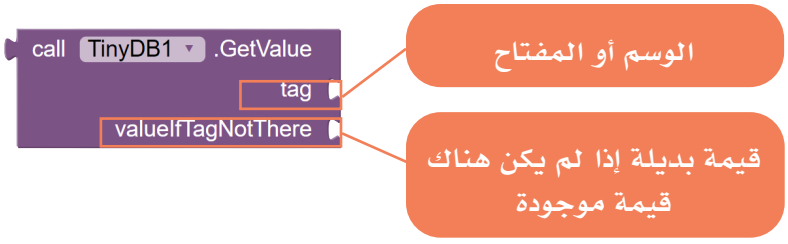
يمتلك كل تطبيق مخزن بيانات خاصاً و واحداً فقط؛ لذا فإن تعدد مكونات TinyDB داخل المشروع لا يعني تعدد المخازن، بل تشترك جميعها في نفس قاعدة البيانات.



• قاعدة البيانات BDyniT تحفظ النصوص فقط، ولا يمكنها حفظ قوائم أو أرقام مباشرة.
وسوف نتعلم لاحقاً كيفية التعامل معها لحفظ قوائم وأرقام.

استرجاع البيانات

تُستخدم الكتلة البرمجية StoreValue لاسترجاع البيانات من قاعدة البيانات.





النشـــاط

مهمة النشاط:

- تعديل تطبيق KuwaitAttractions15 بحيث:

يتم حفظ آخر معلم سياحي تم البحث عنه، ويظهر هذا الاسم عند فتح الشاشة TouristAttractions.

تنفيذ النشاط:

- تشغيل برنامج App Inventor واستدعاء التطبيق KuwaitAttractions15 من مجلد الأنشطة Activity.
- **أولاً : واجهة المصمم Designer:**

1. تحديد شاشة historicalknowledge:
   - إضافة المكون TinyDB من التصنيف storage.
   - تغيير اسم المكون Rename : MainData.
2. تعديل الإجراء Procedures (searchAttraction):
   - التحقق من حالة العثور على المَعلم السياحي في عملية البحث :



- في حالة وجود المعلم السياحي:
  - تحديد المكون MainData ثم من التصنيف Procedures يتم إضافة الكتلة البرمجية:



  - الوسم tag: إضافة النص LastSearched.
  - القيمة valueToStore: إضافة اسم المعلم الذي تم إيجاده لحفظ قيمته.



- عرض بيانات المعلم السياحي من خلال استدعاء الإجراء (DisplayAttraction) وذلك لتحديث واجهة المستخدم؛ حيث يظهر (اسم المعلم، سنة التأسيس، والموقع) .



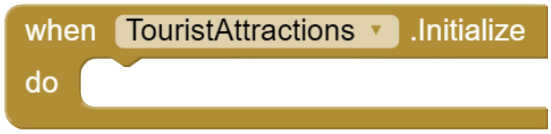
- في حالة عدم العثور على المعلم:
  - ظهور الرسالة: Not Found داخل المكون LblLastSearch. (يتم إضافته للشرط السابق).



- تجميع الكتل البرمجية كالتالي:



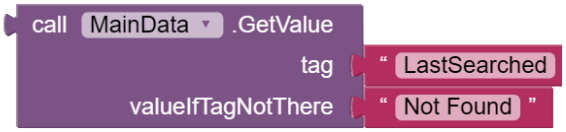
3. عرض اخر مَعلم سياحي تم البحث عنة في كل مرة يتم تحميل الشاشة:
   - تحديد الشاشة TouristAttractions ثم سحب استخدام البرمجية الخاصة بتحميلها.



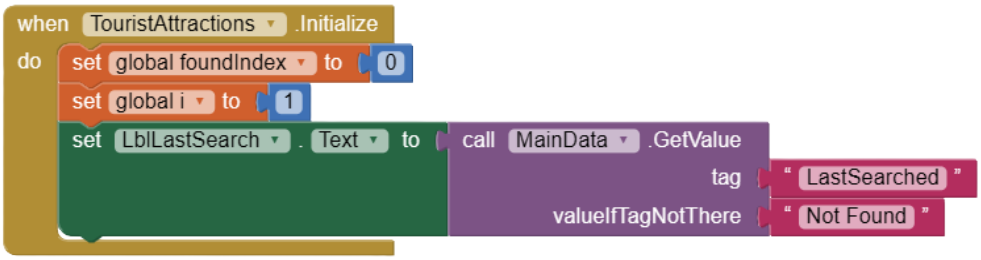
   - تحديد قاعدة البيانات MainData لاستدعاء القيمة المخزنة فيها.



   - الوسم tag: إضافة النص LastSearched. (نفس الاسم الذي تم تعريفة مسبقا في كتلة حفظ البيانات).
   - القيمة valueIfTagNitThere: إضافة النص «Not Found» في حالة لم يكن هناك معلم مخزن.



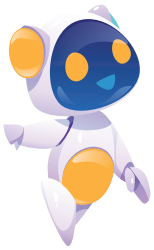
■ الشكل النهائي للكتل البرمجية الخاصة بالشاشة :



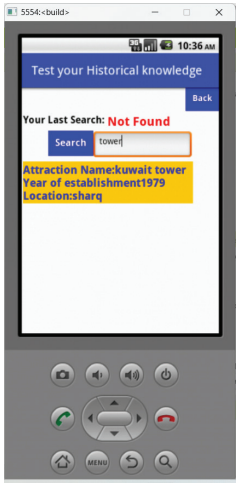


للتأكد من تسجيل قيمة آخر بحث يتم إغلاق المحاكي و إعادة الدخول مرة أخرى على شاشة البحث.

4. حفظ التطبيق باسم KuwaitAttractions16 وعرض التطبيق من خلال المحاكي.









التطبيــق

ورقة عمل (16)

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدع تطبيق HamadStore15.

■ أولاً: واجهة المصمم Designer.

2. حدد شاشة StoreData:

- إضافة المكون TinyDB للاحتفاظ بأخر منتج تم البحث عنه.
- تغير اسم المكون TinyDb1 إلى MainData.

■ ثانياً: واجهة الكتل البرمجية Blocks .

3. اضف الكتل البرمجية اللازمة في Procedures SearchProduct:

■ استخدم الكتلة البرمجية call MainData to store value، بحيث يتم:

- حفظ البحث في MainData.
- ظهور اسم آخر بحث في المكون LblLastSearch.
- وفي حالة عدم وجود نتيجة للبحث تظهر رسالة: Not Found في LblLastSearch.

4. احفظ التعديلات باسم HamadStore16 ثم أعرض التطبيق من خلال شاشة المحاكي.









في وقت فراغك

أنشئ تطبيق يتم في الشاشة الأولى إدخال اسمك في مربع نص، ثم عند الضغط على زر «حفظ الاسم» يتم حفظ الاسم في TinyDB. وعند فتح التطبيق لاحقاً يظهر له مباشرة «مرحبًا يا اسمك» بدون أن يكتبه من جديد، بحيث يكون الاسم مخزّن في TinyDB.

- مكوّنات الواجهة :
  - TextBox: لكتابة الاسم.
  - Button: بعنوان «حفظ الاسم».
  - Label: لعرض رسالة الترحيب.
  - TinyDB: مكوّن غير مرئي.



عبر عن رأيـك



أشرح مفهوم قاعدة البيانات المحلية.	☐	☐	☐
أوضح أهمية TinyDB في حفظ البيانات.	☐	☐	☐
أميز الفرق بين الذاكرة المؤقتة (المتغيرات) والذاكرة الدائمة TinyDB.	☐	☐	☐
أستخدم كتل StoreValue و GetValue لحفظ واسترجاع البيانات.	☐	☐	☐
أطبق TinyDB لحفظ إعدادات أو معلومات المستخدم في تطبيقات واقعية.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

تابع/ قواعد البيانات TinyDB

نواتج التعلم

- شرح كيفية توظيف TinyDB مع القوائم.
- استخدام الكتلة البرمجية list to csv row list لتحويل القوائم إلى نص لحفظها في TinyDB.
- استخدام الكتلة البرمجية list from csv row text لاسترجاع القوائم من TinyDB.
- تنظيم الكتل البرمجية باستخدام إجراءات مخصصة.
- التعامل مع الحالة عندما لا توجد بيانات محفوظة.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

يوفّر برنامج App Inventor إمكانية إنشاء قائمة مفضلات أو سلة مشتريات باستخدام الكتل البرمجية الخاصة بـ CSV لحفظ البيانات داخل TinyDB.

CSV (Comma-Separated Values)

تنسيق نصي بسيط يُستخدم لتخزين بيانات جدولية، مثل : بيانات الجداول أو قواعد البيانات .

- كل سطر يمثل صفًا.
- كل قيمة داخل الصف تمثل عمودًا ويتم فصلها بفواصل (,).
- تنسيق واسع الانتشار ومُستخدم بين العديد من التطبيقات.
- يُستخدم غالبًا لتبادل البيانات بين التطبيقات المختلفة.

استخدام ملفات CSV في برنامج App Inventor

1. استخدام الكتلة البرمجية (قائمة إلى صف) .



- تحوّل هذه الكتلة القائمة إلى نص بالشكل التالي:

"Cap,Cup,T-shirt"

- تحوّل هذه الكتلة قائمة بسيطة (Single List) إلى نص CSV مكوّن من صف واحد فقط.
- يُعتبر كل عنصر في قائمة حقلًا واحدًا داخل الصف.

2. استخدام الكتلة البرمجية (قائمة من صف) .



تحوّل هذه الكتلة نصًا بصيغة CSV إلى قائمة يمكن استخدامها داخل التطبيق.

- إنتاج قائمة من الحقول.
- تقسّم النص اعتمادًا على الفواصل.
- إذا كان النص فارغًا قد تعيد قائمة فارغة.

نشـــاط 1



مهمة النشاط:

تعديل تطبيق KuwaitAttractions16 بحيث:

عند فتح الشاشة يتم حفظ وعرض آخر معلم سياحي تم البحث عنه ، و إنشاء قائمة بالمعالم السياحية المفضلة لعرضها في شاشة historicalknowledge، باستخدام قواعد البيانات وكتل CSV، ويمكن للمستخدم بعد كل عملية بحث إضافة المَعلم إلى القائمة المفضلة لديه.

تنفيذ النشاط :

1. تشغيل برنامج App Inventor واستدعاء التطبيق KuwaitAttractions16 من مجلد الأنشطة Activity.

2. الانتقال إلى واجهة الكتل البرمجية Blocks، وتحديد شاشة TouristAttractions.

- **أولاً : تحديد الزر BtnAddFavorite، وتنفيذ التالي عند الضغط عليه :**

1. سحب الكتلة البرمجية الخاصة بالحدث الخاص بالزر.

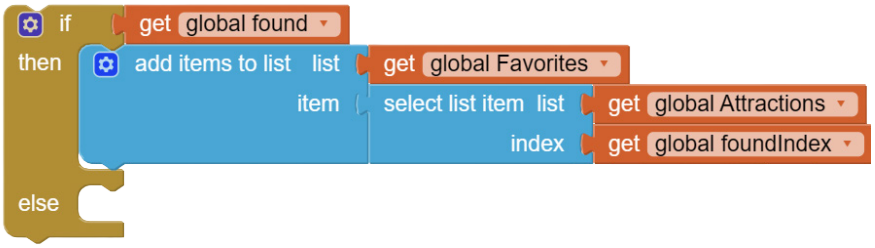


2. التحقق من وجود عملية بحث من خلال استخدام كتلة الشرط if.



- **في حالة ظهور نتيجة البحث :**

3. إضافة المَعلم السياحي (ناتج البحث) إلى القائمة الخاصة بالمتغير Favorites.



4. إضافة المَعلم السياحي (ناتج البحث) إلى القائمة الخاصة بالمتغير Favorites.



5. إظهار رسالة في LblStatus تفيد بإضافة المعلم للمفضلة.



6. استدعاء الإجراء displayFavorites لعرض بيانات المفضلة.



- إذا لم يتم البحث:

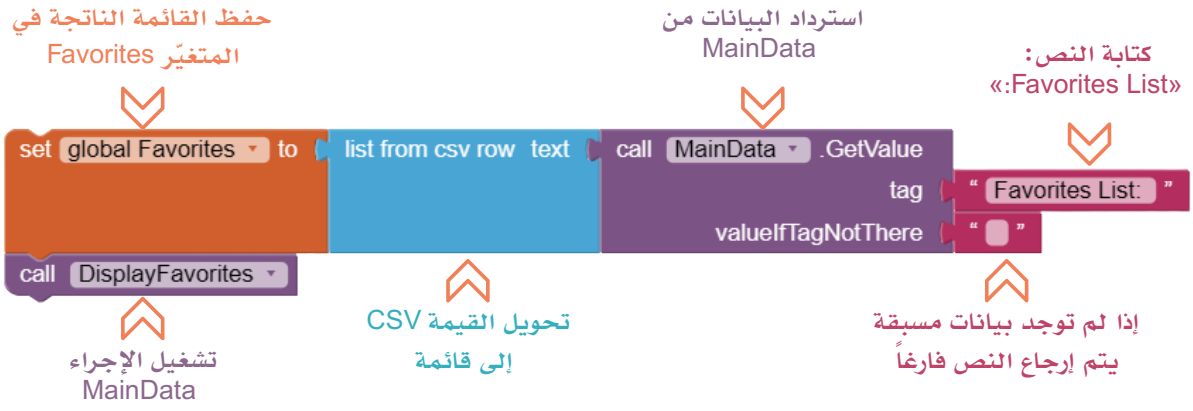
7. تظهر رسالة : " Search first " في المكون LblStatus.



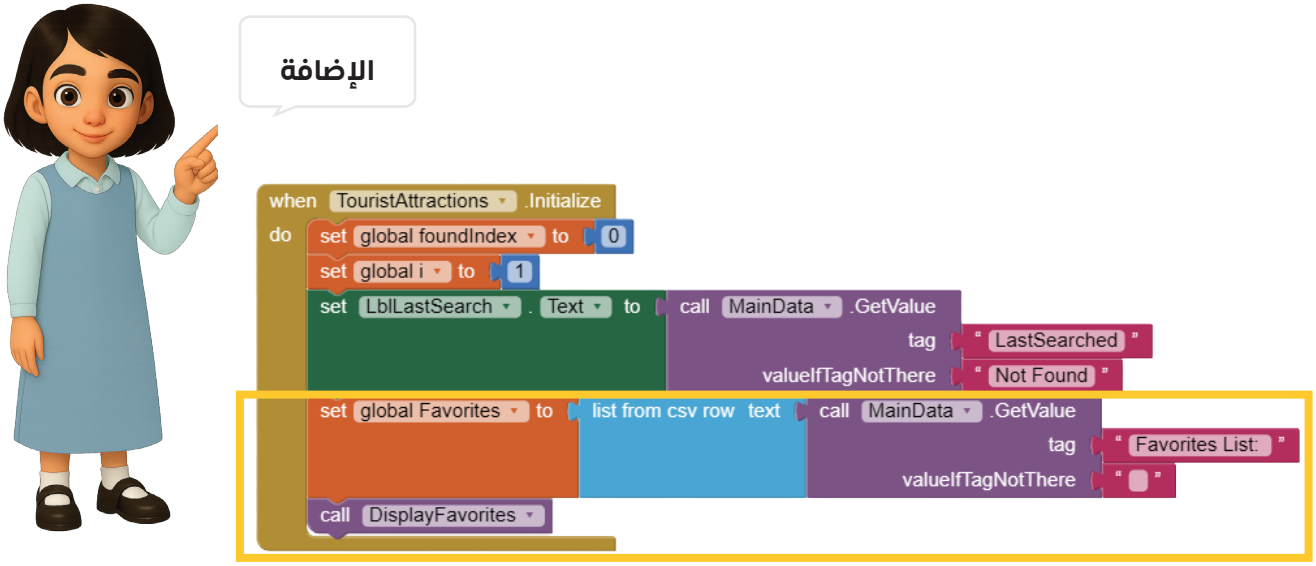
تجميع الكتل البرمجية الخاصة عند الضغط على الزر BtnAddFavorite كالتالي:



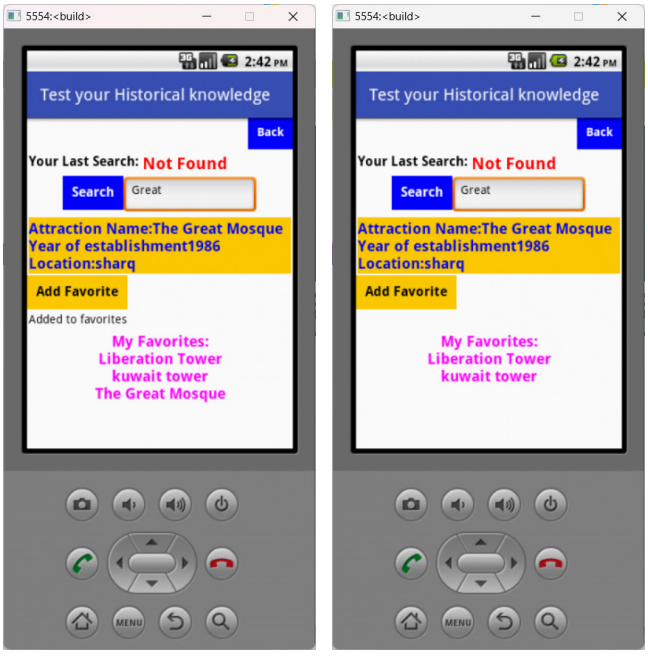
- ثانياً: تحديد الشاشة historicalknowledge ثم استكمال الكتل البرمجية الخاصة بتحميلها:



الكتل البرمجية مجمعة بعد التعديل والخاصة بحدث فتح الشاشة كالتالي:



8. حفظ التطبيق باسم KuwaitAttractions17 وعرض التطبيق من خلال المحاكي.





التطبيـــق

ورقة عمل (17)

المطلـــوب: إنشـــاء قائمـــة بالمنتجـــات المفضّلـــة لـــكل عميـــل تُحفـــظ فـــي قاعـــدة البيانـــات MainData وتظهـــر تلقائيًـــا عنـــد الدخـــول إلـــى شاشـــة البحـــث.

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore16 من مجلد أوراق العمل Workpaper9_2.
2. انتقل إلى واجهة الكتل البرمجية Blocks، ثم حدد شاشة StoreData.
3. عند الضغط على الزر AddToFavorites:
   - يتم تخزين القائمة الناتجة داخل قاعدة البيانات MainData.
   - عند العثور على المنتج في نتائج البحث:
     - تخزين البيانات باستخدام الكتلة: call MainData.StoreValue.
     - تحديد الوسم (tag) الذي تُخزن بداخله البيانات: favoritesProductsList.
     - تحويل قائمة المفضلات إلى نص بصيغة CSV باستخدام: list to CSV raw list.
     - القيمة التي يتم تحويلها إلى CSV هي : favorites (قيمة المتغيّر).
     - إظهار رسالة في LblStatus تفيد بإضافة المنتج للمفضلة.
     - استدعاء الإجراء displayFavorites لعرض بيانات المنتجات المفضلة.
   - إذا لم يتم البحث، تظهر رسالة : «Search first» في المكون LblStatus.
4. عند تشغيل الشاشة StoreData:



   - عرض قائمة المنتجات المفضّلة كالتالي .
     - قراءة البيانات المخزّنة داخل قاعدة البيانات MainData.
     - تحويل النص من CSV إلى قائمة المفضّلة.
     - عرض القائمة المفضلة باستخدام الإجراء displayFavorites.



في وقت فراغك

- أضف شاشة للتطبيق الذي قمت بإنشائه في وقت فراغك السابق ليتم إضافة هواياتك في شكل قائمة من خلال ما تعلمته في درس اليوم.



عبر عن رأيــك



أوظّف TinyDB مع القوائم.	☐	☐	☐
أستخدم الكتلة المناسبة لتحويل القوائم إلى نص.	☐	☐	☐
أستخدم الكتلة المناسبة لاسترجاع القوائم من TinyDB .	☐	☐	☐
أنظم الكتل البرمجية باستخدام إجراءات مخصصة.	☐	☐	☐
أتعامل مع الحالة عندما لا توجد بيانات محفوظة.	☐	☐	☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

تابع/ قواعد البيانات TinyDB

نواتج التعلم

- حذف أو تحديث البيانات المحفوظة في TinyDB.
- استخدام الكتلة البرمجية StoreValue لتحديث قائمة محفوظة (مثل إزالة عنصر من «المفضلة»).
- استخدام الكتلة البرمجية StoreValue مع قيمة فارغة لـ مسح البيانات.
- التعامل مع الحالات الاستثنائية (مثل قوائم فارغة).



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

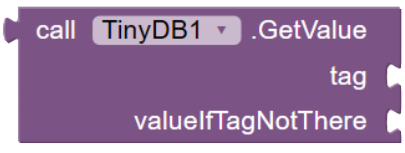
حذف أو تحديث البيانات داخل TinyDB

قاعدة البيانات TinyDB لا تحتوي على كتلة برمجية جاهزة لحذف عنصر واحد حيث أنها تخزّن البيانات بشكل كامل تحت وسم Tag.

حذف وتعديل بيانات TinyDB:

- **الخطوة الأولى: استدعاء القائمة المخزّنة**

استدعاء القائمة من TinyDB باستخدام الكتلة:

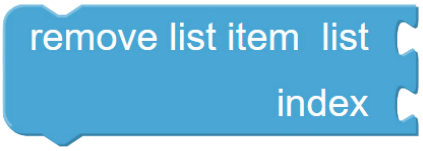


- تحديد الوسم (Tag) الذي يحتوي على القائمة.
- بعد الاستدعاء، تصبح القائمة داخل متغيّر ويمكن تعديلها بسهولة.

الخطوة الثانية: تعديل القائمة

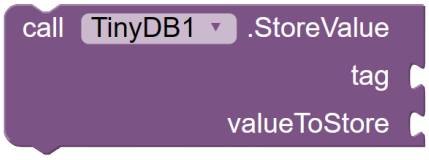
بعد أن تصبح القائمة داخل المتغيّر، يمكن استخدام أدوات App Inventor :

- إزالة عنصر واحد من القائمة باستخدام الكتلة:



حيث يمكن حذف عنصر محدد من القائمة عن طريق رقم الفهرس أو عبر البحث عن قيمته.

الخطوة الثالثة: حفظ القائمة المعدّلة من جديد



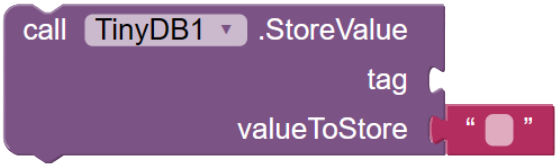
بعد تعديل القائمة (حذف أو تحديث)، يتم إعادة تخزينها باستخدام الكتلة:

- نفس الوسم Tag.
- والقيمة الجديدة هي القائمة بعد التعديل.

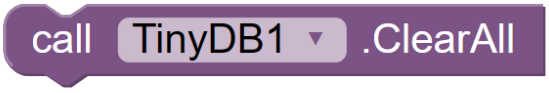
حذف البيانات في TinyDB:

حذف بيانات Tag واحد فقط.

استخدام valueToStore مع قيمة فارغة (حذف المحتوى مع الإبقاء على اسم الـ Tag مسجلاً في القائمة ولكنه «فارغ»).



حذف كل البيانات في TinyDB .





نشـــاط 1

مهمة النشاط:

استكمال نشاط الدرس السابق تطبيق KuwaitAttractions17 بحيث يتم:

- حـذف عنصـر مـن قائمـة المعالـم السـياحية المفضلـة فـي قاعـدة البيانـات، ثـم تحديث القيمـة المخزنـة بعـد التعديل.
- وحـذف جميـع البيانـات المرتبطـة بوسـم معيّـن باسـتخدام StoreValue مـع قيمـة فارغـة.

تنفيذ النشاط :

1. تشـغيل برنامـج App Inventor واسـتدعاء التطبيـق KuwaitAttractions17 مـن مجلـد الأنشـطة Activity.
2. الانتقال إلى واجهة الكتل البرمجية Blocks، وتحديد شاشة TouristAttractions.

- أولاً: حذف عنصر من القائمة عند الضغط على زر الحذف: BtnRemoveFavorite.



- تهيئة المتغيّر i = 1.
- تهيئة المتغيّر found = false.

3. إنشاء حلقة while تعمل عندما:

- i =< طول القائمة و found = false (لم يتم إيجاد المعلم)

يتوقف البحث يتوقف فور العثور على المعلم السياحي.



4. مقارنة تطابق النصوص:

- قراءة النص المُدخل من المكون TextRemove واستخدام trim لإزالة المسافات.
- مع العنصر رقم i في القائمة.



5. عند التطابق:

- حذف العنصر من القائمة.
- تغيير قيمة المتغير found إلى true لإيقاف الحلقة التكرارية.

تذكـــر

- بعـد الحـذف، لا نزيـد i لأن العنصـر التالي يتحـرك تلقائيًا إلى مـكان العنصر المحذوف..

6. عند عدم التطابق:

- الانتقال إلى العنصر التالي i+1.



7. بعد انتهاء الحلقة، يتم التحقق من النتيجة:

- إذا كان المتغير found = true (أي وُجد المَعلم).
  - تحديث القائمة MainData.
  - تحويل القائمة إلى CSV.
  - تخزين القائمة المحدثة.
  - عرض رسالة تأكيد الحذف: "Removed from favorites".

- إذا كان المتغير found = false.
  - عرض رسالة : Not found in favorites.



- استدعاء الإجراء displayFavorites لعرض القائمة المحدثة.

تكون الكتل البرمجية بعد التجميع كما يلي:

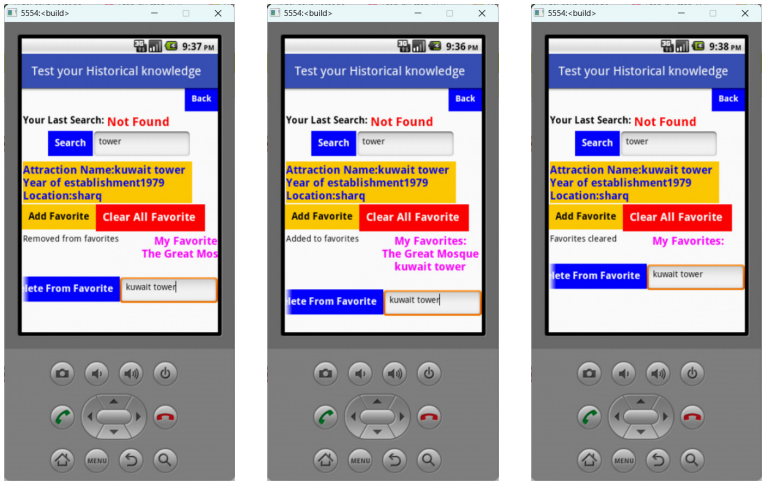


- ثانياً: حذف جميع البيانات في المكون MainData عند الضغط على زر الحذف: BtnClearFavorite .

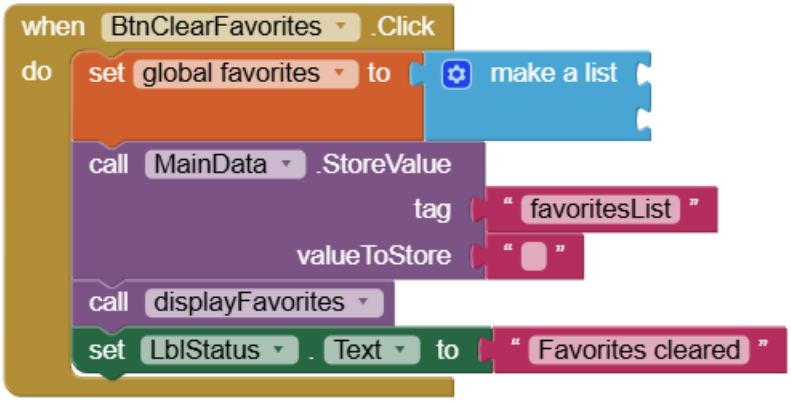
1. تهيئة المتغيّر favorites : قائمة فارغة
2. استخدام كتلة : call MainData.StoreValue.
   - الوسم tag : المستخدم لتخزين القائمة.
   - القيمة valueToStore: "" - خالية.

3. استدعاء displayFavorites لعرض قائمة فارغة.
4. في المكون LblStatus عرض رسالة: "Favorites cleared"

يكون الكتل البرمجية بالشكل التالي :



5. حفظ التطبيق باسم KuwaitAttractions18 وعرض التطبيق من خلال المحاكي.





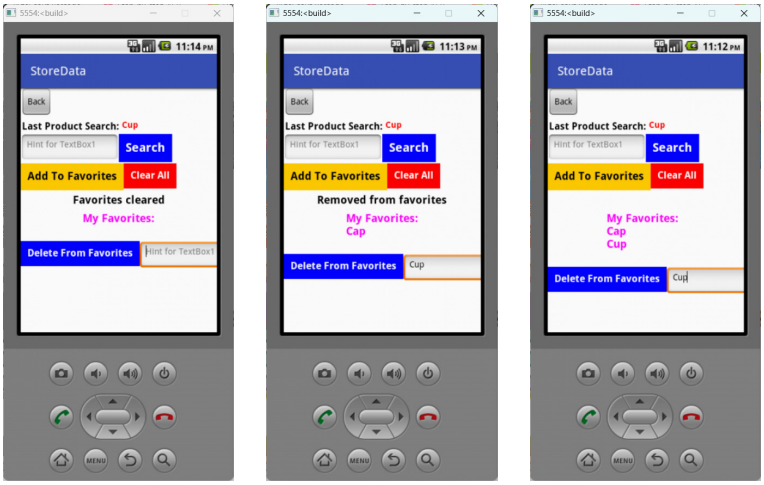
التطبيـــق

ورقة عمل (18)

المطلـوب: حـذف جميـع البيانـات فـي المكـون MainData تحـت وسـم معيـن tag باسـتخدام call MainData.StoreValue.

من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدعِ التطبيق HamadStore17 من مجلد أوراق العمل Workpaper9_2.
2. انتقل إلى واجهة الكتل البرمجية Blocks، ثم حدد شاشة StoreData.
3. أضف الكتلة البرمجية المناسبة للزر BtnRemoveFavorite لحذف أحد عناصر القائمة المفضلة .
4. أضف الكتل البرمجية المناسبة للزر BtnClearFavorites لحذف جميع العناصر من القائمة المفضلة .
5. احفظ التعديلات باسم HamadStore18 ثم أعرض التطبيق من خلال شاشة المحاكي.





في وقت فراغك

- في شاشة الهوايات التي أضفتها للتطبيق الذي قمت بإنشائه في وقت فراغك السابق احذف بعض الهوايات وأضف البعض وشاهد النتيجة وشاركها مع معلمك.



عبر عن رأيــك



أحذف أو أحدّث البيانات المحفوظة في TinyDB.		
استخدم StoreValue لتحديث قائمة محفوظة (مثل إزالة عنصر من «المفضلة»).		
أستخدم valueToStore مع قيمة فارغة لـمسح البيانات.		
أتعامل مع الحالات الاستثنائية (مثل قوائم فارغة).		



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

إنشاء شاشة رئيسية

نواتج التعلم

- تنظيم مكونات الشاشة الرئيسية باستخدام التنسيق الأفقي والرأسي.
- توظيف الأزرار المناسبة للتنقل بين الشاشات المختلفة في التطبيق.
- ضبط خصائص المكوّنات في الشاشة الرئيسية مثل الصور، وأبعادها، وألوان الخلفية، وحجم الخط.
- استخدام الكتلة البرمجية open another Screen screen name للانتقال بين الشاشات.
- إدراج اسم الشاشة داخل الكتلة النصية.
- تنفيذ خطوات ربط أزرار الشاشة الرئيسية بالشاشات الأخرى في التطبيقات.
- عرض التطبيق في المحاكي للتحقق من صحة عمل الشاشة الرئيسية وأزرار التنقل بين الشاشات.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــــم

تنظيم الشاشة الرئيسية في التطبيق

- إضافة الازرار المناسبة.
- إضافة الصور والتحكم في أبعادها.
- اختيار ألوان مناسبة للمكونات.
- تحديد حجم الخط بما يناسب وضوح الشاشة.

إنشاء شاشة رئيسية

- استخدام الكتلة البرمجية: open another screen screen name للانتقال بين الشاشات.



- إدراج اسم الشاشة داخل الكتلة البرمجية النصية A Text String.
- ترتيب المكونات باستخدام كل من VerticalArrangement او HorizontalArrangement من تصنيف Layout.



النشـــاط

مهمة النشاط:

- إضافة الكتل البرمجية المناسبة لشاشة MainScreen في تطبيق UserInterface للتنقل بين الشاشات التي تم إنشائها في التطبيق السابق KuwaitAttractions.

تنفيذ النشاط:

- تشغيل برنامج App Inventor واستدعاء التطبيق KuwaitAttractions من مجلد الأنشطة Activity.



في التطبيق السابق KuwaitAttractions: تم إنشاء 5 شاشات مختلفة كما في الجدول التالي:

اسم الشاشة	الهدف منها
Screen1	واجهه رئيسية للتطبيق.
TouristAttractions	تغير ألوان الخلفية الشاشة .
TicketStatus	نظام حجز التذاكر للأبراج.
Historicalknowledge	عرض معلومات متنوعة عن المعالم السياحية وأماكنها.
VisitorRecord	عرض أسماء الزائرين وأنواع الزيارة وتاريخها.

واجهة الكتل البرمجية Blocks:

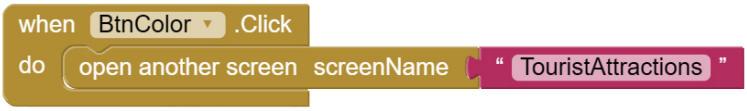
- اختيار شاشة MainScreen.

1. إضافة الكتلة البرمجية المناسبة لمكونات الشاشة لتنفيذ التالي:

- زر BtnStartScreen: الانتقال إلى شاشة Screen1.



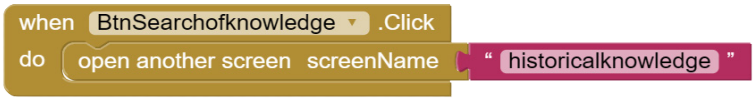
- زر BtnColor: الانتقال إلى شاشة TouristAttractions.



- زر BtnBooking: الانتقال إلى شاشة TicketStatus.



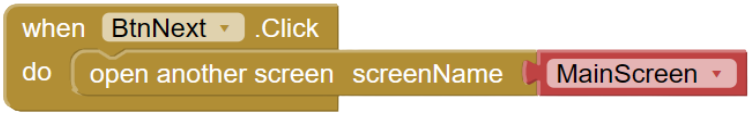
- زر BtnSearchofknowledge: الانتقال إلى شاشة historicalknowledge.



- زر BtnVisitorCard: الانتقال إلى شاشة visitorRecord.

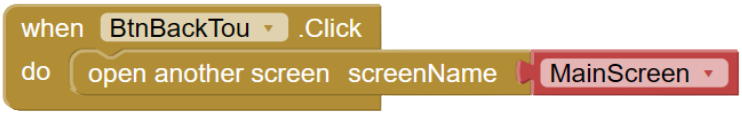


- إضافة زر لجميع الشاشات للرجوع إلى الشاشة الرئيسية MainScreen التالي:
  - إضافة الكتلة البرمجية لزر BtnNext في شاشة Screen1 للانتقال إلى شاشة MainScreen.

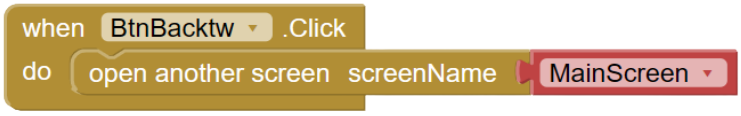


إنشاء شاشة رئيسية

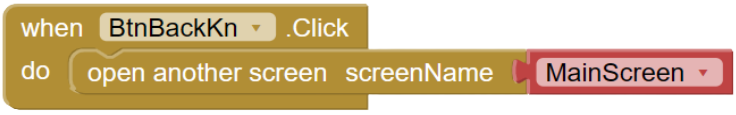
- إضافة الكتلة البرمجية للزر BtnBackTou في شاشة TouristAttractions للانتقال إلى شاشة MainScreen.



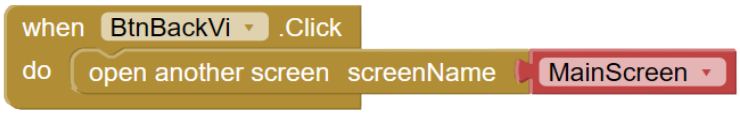
- إضافة الكتلة البرمجية للزر BtnBacktw في شاشة Towertiket للانتقال إلى شاشة MainScreen.



- إضافة الكتلة البرمجية للزر BtnBackKn في شاشة historicalknowledge للانتقال إلى شاشة MainScreen.



- إضافة الكتلة البرمجية للزر BtnBackVi في شاشة visitorRecord للانتقال إلى شاشة MainScreen.



- حفظ التطبيق باسم UserInterface وعرض التطبيق من خلال المحاكي.





التطبيــق

ورقة عمل (19)

- من خلال دراستك لبرنامج App Inventor نفذ الخطوات التالية:

1. استدع تطبيق HamadInterFace من مجلد WorkPapers9_2.
2. انتقل الى شاشة MainPage.

- عزيزي المتعلم الشاشات التى تم تصميمها فى التطبيق السابق HamadStore:

اسـم الشـاشـة	الهـدف منهـا
Screen1	واجهه رئيسية للتطبيق.
Categories	تغير ألوان خلفية الشاشة.
PurchaseRecord	معلومات عن المشترين وتاريخ الشراء.
SaleItem	مبيعات المنتجات.
StoreData	بيانات المنتجات و وأسعارها.

واجهة الكتل البرمجية Blocks:

- اختر الشاشة MainPage.

3. إضافة الكتلة البرمجية المناسبة لمكونات الشاشة لتنفيذ التالي:
   - زر BtnStartScreen: الانتقال إلى شاشة Screen1.
   - زر BtnColor: الانتقال إلى شاشة Categoried.
   - زر BtnRecord: الانتقال إلى شاشة PurchaseRecord.
   - زر BtnSale: الانتقال إلى شاشة SaleItem.
   - زر BtnDataStore: الانتقال إلى شاشة StoreData.
4. حفظ التطبيق باسم HamadInterFace1 وعرض التطبيق من خلال المحاكي.



عبر عن رأيـك



أنظم مكونات الشاشة الرئيسية.	☐	☐ ☐
أوظـف الأزرار المناسـبة للتنقـل بيـن الشاشـات المختلفـة فـي التطبيـق.	☐	☐ ☐
أضبط خصائص المكوّنات في الشاشة الرئيسية.	☐	☐ ☐
أستخدم الكتلة البرمجية للانتقال بين الشاشات.	☐	☐ ☐
أنفـذ خطـوات ربـط أزرار الشاشـة الرئيسـية بالشاشـات الأخـرى فـي التطبيق .	☐	☐ ☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

أداة دمج التطبيقات AI2MergerApp

نواتج التعلم

- شرح مفهوم أداة AI2MergerApp في MIT App Inventor وأهميتها.
- دمج شاشة من تطبيق مستقل في تطبيق رئيسي باستخدام أداة AI2MergerApp.
- الحفاظ على وظائف الشاشة المدمجة (الأزرار، المكونات، الأحداث).
- تطبيق أداة AI2MergerApp لدمج واجهة مستخدم موحدة في مشاريع متعددة.
- حل مشكلات تعارض الأسماء للمكونات والشاشات أثناء عملية دمج التطبيقات.
- استخدام المحاكي في App Inventor لاختبار التطبيق النهائي بعد الدمج والتأكد من عمل الشاشات المدمجة.



ملفات أوراق العمل
ملفات مصادر التعلم



الاستكشاف





التعلـــم

أداة AI2MergerApp

تُعدّ أداة دمج التطبيقات في App Inventor أداةً بالغة الأهمية عند تطوير التطبيق ضمن فريق. فهي تسمح لعدة مطورين بالعمل على شاشات مختلفة من التطبيق، ثم دمجها معًا في مشروع واحد.

أداة دمج التطبيقات AI2MergerApp

لاحظ

- الحد الأقصى الموصى به لعدد الشاشات في تطبيق App Inventor هو 10 شاشات.
- يجب أن يكون لكل مشروع Screen1 واحد فقط.
- يمكن للفرق الأخرى ترك Screen1 فارغة وتطوير شاشات إضافية.
- إذا وُجد Screen1 أخرى، ستسمح أداة الدمج بإعادة تسمية الشاشة أثناء الدمج.

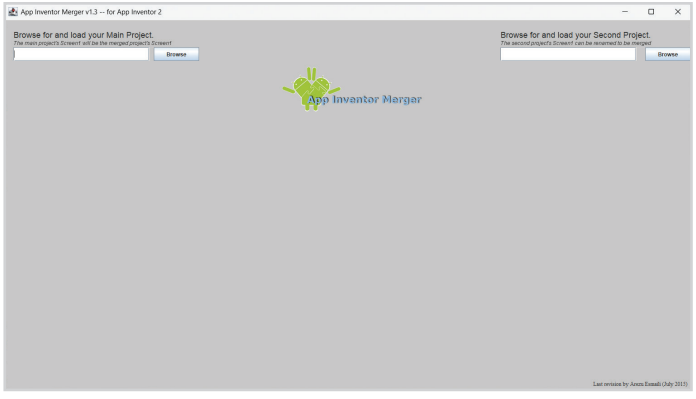
الفكرة العامة لعمل AI2MergerApp

- **التنسيق المسبق:** يمكن للمطور استدعاء شاشة صممها زميله بشرط معرفة الاسم الدقيق الذي أطلقه عليها.
- **توحيد الأسماء:** إذا كانت هناك شاشتان تستخدمان نفس قاعدة البيانات، يجب أن يكون لقاعدة البيانات نفس الاسم في كل الشاشات وكذلك التأكد من Tag المستخدم سواء في حفظ او استدعاء البيانات أن يكون موحد في الكتل البرمجية لضمان عملها بشكل صحيح بعد الدمج.
- كل فريق يطوّر شاشة أو أكثر ضمن مشروع منفصل.
- عند الدمج:
  - يتم اختيار التطبيق الرئيسي.
  - ثم اختيار التطبيق الفرعي الذي يحتوي على الشاشة المراد دمجها.
  - يتم تحديد الشاشات والأصل .
  - ثم يتم الضغط على Merge لدمج الشاشات داخل التطبيق النهائي.



خطوات تشغيل وعمل أداة AI2MergerApp في برنامج App Inventor

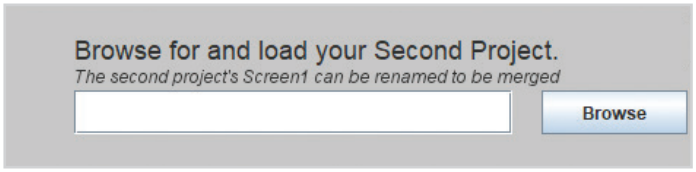
1. تشغيل الأداة AI2MergerApp بالضغط عليها، تظهر الشاشة التالية:



2. اختيار التطبيق الرئيسي من خلال Browse for and load your Main Project

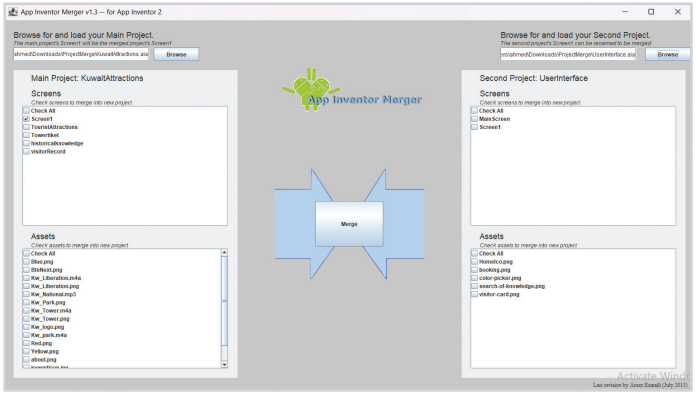


3. اختيار التطبيق الفرعي من خلال Browse for and load your Second Project

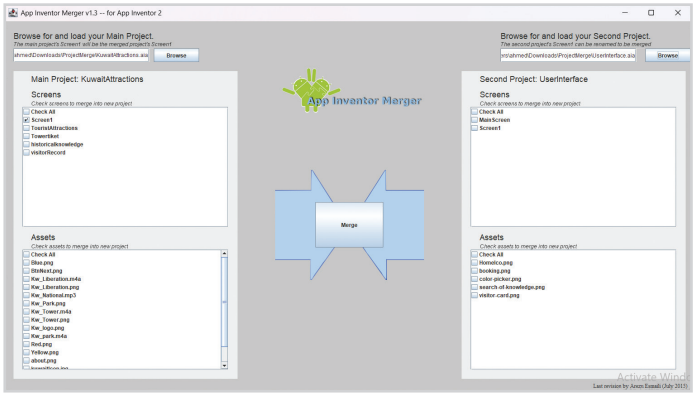


بعد اختيار المشروع الرئيسي و الفرعي تظهر الشاشة كما يلي:

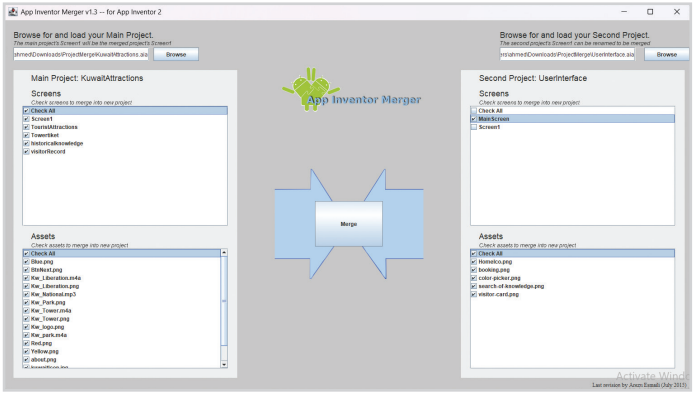




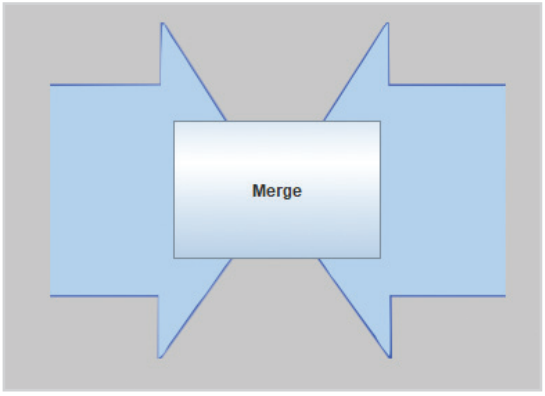
4. تحديد كل الشاشات Screens وكذلك الأصول Assets في التطبيق الرئيسي وذلك بالضغط على Check All في Screens و Assets.



5. تحديد الشاشات Screens و الأصول Assets الخاصة بها في التطبيق الفرعي كما بالشكل التالي:

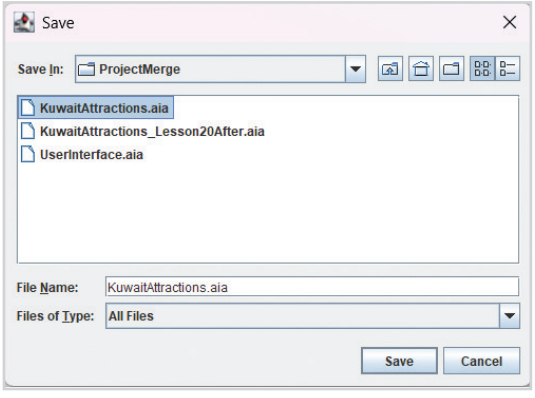


6. الضغط على زر Merge لإتمام عملية الدمج.



7. تظهر شاشة حفظ التطبيق، يتم كتابة اسم التطبيق الجديد ومكان الحفظ ثم الضغط على زر Save.





8. فتح المشروع النهائي في App Inventor واختباره باستخدام المحاكي.



النشـــاط

مهمة النشاط:

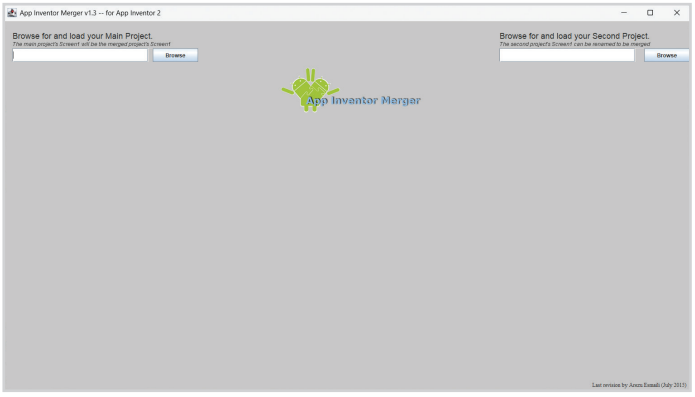
استكمالا لما تم تنفيذه في الدروس السابقة المطلوب:

- دمـــج واجهـــة تطبيـــق KuwaitAttractions داخـــل تطبيـــق UserInterface ودمـــج شاشـــة MainScreen فـــي تطبيـــق KuwaitAttractions باســـتخدام أداة AI2MergerApp.
- حفظ المشروع باسم KuwaitAttractionsFinal.

تنفيذ النشاط:

- تشـــغيل برنامـــج App Inventor واســـتدعاء التطبيـــق KuwaitAttractions18 مـــن مجلـــد الأنشـــطة Activity.

1. استدعاء أداة AI2MergerApp لتظهر الشاشة الرئيسية للأداة:



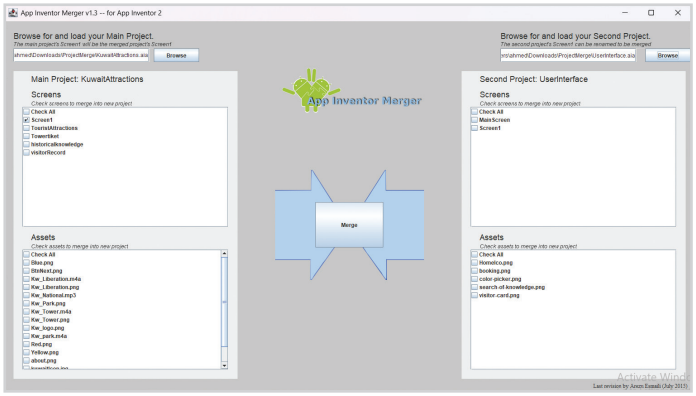
2. اختيار التطبيق الرئيسي من خلال Browse for and load your Main Project.

   اسم التطبيق الرئيسي KuwaitAttractions.

3. التأكد من اختيار Check All في الشاشات Screens.

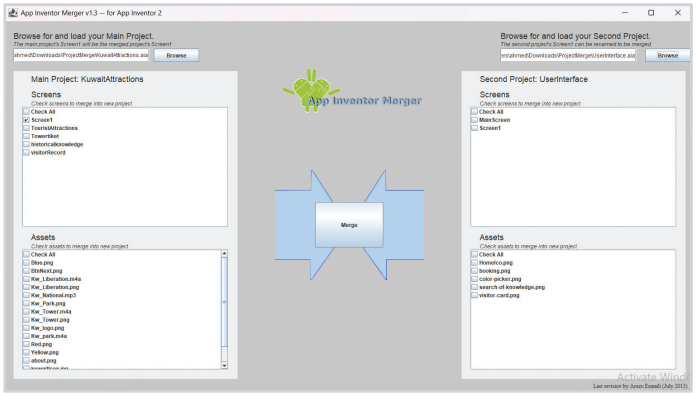
4. التأكد من اختيار Check All في الأصول Assets.

5. اسم التطبيق الفرعي UserInterface.



6. اختيار الشاشة MainScreen.

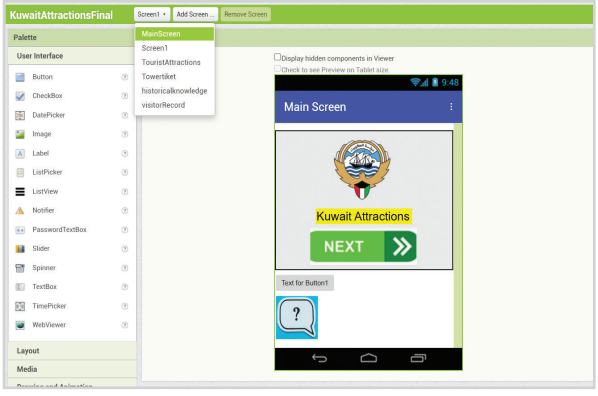
7. تحديد كل Assets ليتم دمجها في عملية Merge.



8. الضغط على زر Merge.

9. تظهر شاشة حفظ التطبيق بعد الدمج اكتب اسم التطبيق و مكان الحفظ و اضغط زر Save.

10. حفظ التطبيق باسم KuwaitAttractionsFinal.



ظهور شاشة MainScreen

11. استدعاء التطبيق KuwaitAttractionsFinal وعرضه من خلال المحاكي.



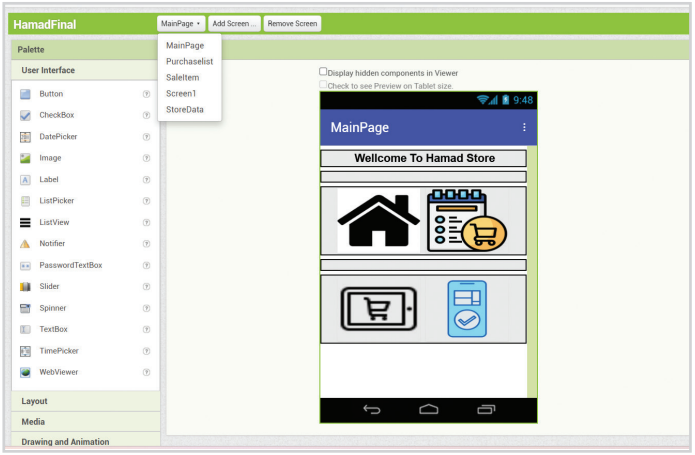


التطبيــق

ورقة عمل (20)

استكمالا لما تم تنفيذه في الدروس السابقة، المطلوب:

1. ادمج واجهة تطبيق HamadStore داخل تطبيق HamadInterFace.
2. ادمج الشاشة MainPage: من تطبيق HamadStore باستخدام أداة AI2MergerApp.
3. احفظ التطبيق النهائي باسم HamadStoreFinal.



4. اعرض التطبيق في المحاكي.





عبر عن رأيـك



أشرح مفهوم أداة AI2MergerApp في MIT App Inventor.	☐	☐ ☐
أدمـج شاشـة مـن تطبيـق مسـتقل (فـي تطبيـق رئيسـي) باسـتخدام أداة AI2MergerApp.	☐	☐ ☐
أطبـق أداة AI2MergerApp لدمـج واجهـة مسـتخدم موحـدة فـي مشـاريع متعـددة.	☐	☐ ☐
أسـتخدم المحاكـي فـي App Inventor لاختبـار التطبيـق النهائـي بعـد الدمـج والتأكـد مـن عمـل الشاشـات المدمجـة.	☐	☐ ☐



ملاحظات المعلم

اليوم	التاريخ	الملاحظات

ملاحظات ولي الأمر

الوحدة الثانية: المنجات الرقمية

وحدة المشروعات
Projects Unit

1. فكرة وأهداف المشروع.
2. مهارات المشروع.
3. إجراءات تنفيذ المشروع.
4. تطبيق Bright Moments In Kuwait's History .
5. نماذج لأفكار مشروعات متنوعة.

وحدة المشروعات Projects Unit

نواتج التعلم

- فكرة و أهداف المشروع.
- مهارات المشروع.
- إجراءات تنفيذ المشروع.
- تطبيق Bright Moments In Kuwaits History.
- نماذج لأفكار مشروعات متنوعة.



ملفات أوراق العمل
ملفات مصادر التعلم



وحدة المشاريع

إن الهدف الأساسي من إنتاج المشروع هو الاستفادة من المهارات التي تم دراستها وتطبيقها من خلال أوراق العمل بالإضافة إلى تنمية مهارات العمل الجماعي التعاوني، والقدرة على تجميع الوسائط اللازمة لإنتاج المشروع وفق أسس تصميم وبناء تطبيقات الهواتف الذكية والقدرة على تنمية الابتكار.

فكرة المشروع

تصميم مشروع تقني تعليمي بسيط لحل مشكلة أو تنفيذ فكرة مفيدة بطريقة إبداعية على أن تكون واضحة وقابلة للتطبيق.

- سجل فكرة المشروع بعد مناقشة أعضاء الفريق:

أهداف المشروع

يهدف المشروع إلى تنمية المهارات التقنية والإبداعية من خلال تصميم وتنفيذ فكرة تقنية تعليمية بطريقة منظمة وتعاونية.

- سجل أهداف مشروعك:

التوظيف الواقعي في الحياة العملية

يساهم المشروع في ربط التقنية بواقع الحياة من خلال توظيف تطبيقات الهواتف الذكية في حل مشكلات حقيقية أو تحسين ممارسات يومية بطريقة مبتكرة

■ سجل كيف ستوظف المشروع الخاص بك في الحياة العملية:

مهارات المشروع

يهدف المشروع إلى تنمية مهارات المتعلمين التقنية مثل توظيف تطبيقات الهواتف الذكية في حل مشكلات حقيقية، إلى جانب المهارات الشخصية كالتعاون وحل المشكلات، وذلك من خلال تنفيذ فكرة عملية لإكسابهم خبرات واقعية تعزز جاهزيتهم للمستقبل، وتساعدهم على اكتساب كفاءات تقنية تعزز قدرتهم على الابتكار والتطبيق العملي.

■ حدد المهارات المستخدمة في المشروع:

	المجال	المهارات الفرعية المستهدفة
□	المهارات الرقمية والتقنية	□ تصميم تطبيق هاتف ذكي. □ تصميم واجهات أو للتطبيق. □ تحليل ومعالجة بيانات التطبيق.
□	التفكير الناقد وحل المشكلات	□ تحليل المشكلة وتحديد عناصرها. □ تجريب الحلول واختيار الأنسب. □ التحقق من دقة النتائج وتعديل الأخطاء.
□	مهارات التوثيق والبحث العلمي	□ تسجيل مراحل المشروع بالصور والنصوص. □ جمع معلومات من مصادر موثوقة. □ كتابة المراجع العلمية.
□	مهارات العمل الجماعي	□ التعاون وتقسيم المهام. □ تحمل المسؤولية والالتزام بالوقت. □ اتخاذ قرارات جماعية.
□	مهارات العرض والتواصل	□ إعداد عرض بصري فعّال (شرائح، فيديو، أدلة، تقارير). □ شرح الفكرة بلغة واضحة ومقنعة. □ التفاعل مع الجمهور والرد على الأسئلة.

مهام فريق العمل

تقسم المهام بين أعضاء الفريق على النحو التالي:

اسم المتعلم	المهمة	الوصف
	قائد الفريق	المسؤول عن إدارة عمل الفريق وتجميع الملفات والمستندات المطلوبة وتقديمها للمعلم.

مراحل تنفيذ المشروع : (مخطط زمني مقترح)

المرحلة	الاسبوع	الناتج المتوقع
الأسبوع 1	التخطيط والتحليل	☐ اختيار فكرة المشروع. ☐ تحديد المشكلة والأهداف. ☐ إعداد وثيقة المتطلبات الأولية.
الأسبوع 2	خطة المشروع والتوزيع	☐ توزيع المهام على أعضاء الفريق. ☐ إعداد خطة تنفيذ المشروع أسبوعياً.
الأسبوع 3	التصميم والتوثيق الأولي	☐ تصميم الواجهة. ☐ إعداد خطة العمل بالتفصيل.
الأسبوع 4	البرمجة والتنفيذ الأولي	☐ البدء في تنفيذ المشروع عملياً. ☐ تجربة الأدوات البرمجية والتقنية.

الأسبوع 5	الاختبار والتحسين	☐ مراجعة الأخطاء البرمجية. ☐ اختبار الأداء. ☐ إجراء التعديلات والتحسينات النهائية.
الأسبوع 6	التوثيق النهائي والعرض	☐ كتابة التقرير النهائي. ☐ إعداد العرض التقديمي. ☐ التدرب على العرض وتقديمه أمام الصف.

يختلف عدد أسابيع المشروع والمراحل المختلفة وفق خطة توزيع المنهج.

التوثيق العلمي للمشروع

1. المراجع:

- الالتزام بطريقة علمية لتوثيق المراجع على سبيل المثال نظام APA.

2. عرض ومشاركة المشروع

- يعرض الفريق المشروع ويناقشه مع المعلم والمتعلمين، ومشاركته على منصة التعلم عن بعد Teams.

3. الاستمرار في التعلم

- التوسع في الاطلاع على المستجدات التقنية وتنمية قدرات المتعلم البرمجية من خلال الدورات التدريبية والمنتديات الحاسوبية.

متطلبات العرض التقديمي:

إنشاء عرض تقديمي يشتمل على الجوانب التالي:



1. عرض بيانات المدرسة وأعضاء الفريق.
2. الترحيب بالمعلم وزملائه المتعلمين.
3. عرض مقدمة عن فكرة المشروع موضحاً الهدف والمشكلة والفئة المستفادة والحل.
4. عرض النموذج البرمجي.
5. عرض المراجع .
6. فتح باب الأسئلة والمناقشة.
7. الختام.



مشروع تطبيق Bright Moments In Kuwait's History

هـذا المشـروع هـو توظيـف لـكل المهـارات التـي تـم تعلمهـا خلال هـذا الكتـاب، ولا يتـم اسـتخدامه كمشـروع للمتعلميـن، وهـو تطبيـق يعتمـد التعـرف علـى لحظـات تاريخـه لهـا أثـر فـي تاريـخ دولـة الكويـت الحبيبـة.

الأهداف التقنية للمشروع

- إعـداد واجهـة تطبيـق رئيسـية تتضمـن شـعار دولـة الكويـت وعنـوان التطبيـق وزر انتقـال للشاشـة التاليـة.
- توظيـف أداة AI2MergerApp فـي إنشـاء واجهـة مسـتخدم مسـتقلة ودمجهـا فـي مشـروع رئيسـي لتسـريع عمليـة التطويـر وتقليـل الأخطـاء.
- تصميـم شاشـة رئيسـية MainScreen للتنقـل بيـن عـدة شاشـات فـي التطبيـق باسـتخدام أزرار مرتبطـة بكتـل برمجيـة لفتـح الشاشـات.
- توظيـف مكونـات Label و Button و HorizontalArrangement فـي تنظيـم واجهـة المسـتخدم وتحسـين شـكل الأزرار وإضافـة الصـور المناسـبة لهـا.
- إنشاء شاشة KwFlag_Color لعرض ألوان علم الكويت باستخدام أزرار تمثل الألوان.
- توظيف مكونات Label و Image في عرض أبيات الشعر المرتبطة بألوان علم الكويت.
- تغيير لون مكوّن Label برمجياً بناءً على لون الزر الذي يضغطه المستخدم.
- إضافة زر رجوع للانتقال من شاشة KwFlag_Color إلى الشاشة MainScreen.
- إنشـاء شاشـة Kwknowledge لعـرض لحظـات تاريخيـة مضيئـة فـي تاريـخ دولـة الكويـت مـع السـنوات المرتبطـة بـكل حـدث.
- توظيف DatePicker في عرض التاريخ الحالي داخل التطبيق.
- تنفيـذ عمليـة بحـث داخـل قائمـة الأحـداث باسـتخدام متغيـرات وقوائـم وإجـراء searchAction، ثـم عـرض النتائـج فـي Label مخصـص.

- توظيف TinyDB لحفظ الأحداث المفضلة للمستخدم، وإضافة حدث للمفضلة باستخدام زر خاص، وعرض القائمة عبر إجراء displayFavorites عند فتح الشاشة.
- إنشاء شاشة ScreenSound لتشغيل وإيقاف السلام الوطني الكويتي والتنقل بينها وبين الشاشة الرئيسية.
- تنظيم منطق التطبيق باستخدام الإجراءات Procedures، والمتغيرات، والقوائم لإدارة البحث، العرض، والحفظ.
- استثمار أدوات الذكاء الاصطناعي التوليدي في استكشاف معالم إضافية من معالم دولة الكويت والبحث عن تاريخ إنشائها وتضمينها في التطبيق.

الأهداف التربوية للمشروع

- اكتساب مهارات البرمجة باستخدام MIT App Inventor وتطبيق منطق البرمجة بالكتل.
- تنمية مهارات تصميم واجهات المستخدم وتنظيم المكونات البصرية داخل التطبيق.
- تنمية التفكير المنطقي وحل المشكلات من خلال استخدام المتغيرات والقوائم والإجراءات.
- تعزيز الانتماء الوطني عبر التعرّف إلى أحداث بارزة في تاريخ دولة الكويت.
- تطوير مهارات البحث والتحقق من صحة المعلومات التاريخية.
- تنمية مهارات إدارة البيانات بتوظيف TinyDB لحفظ المعلومات واسترجاعها.
- تنمية التذوق اللغوي والأدبي من خلال ربط ألوان علم الكويت بأبيات شعر عربية ذات دلالات وطنية.
- دعم مهارات التعلم الذاتي باستثمار أدوات الذكاء الاصطناعي التوليدي.
- تنمية مهارات العمل بالمشروعات من خلال تنفيذ مشروع متكامل يدمج المعرفة الوطنية بالمهارات التقنية في بيئة MIT App Inventor.

مكونات التطبيق

تصميم تطبيق تعليمي رقمي يعمل على الهواتف الذكية، يعرض لحظات مضيئة في تاريخ دولة الكويت بأسلوب مبسّط وتفاعلي
يركّز التطبيق على تعريف المتعلم بأهم الأحداث الوطنية مثل الدستور، التحرير، الاستقلال، وإطفاء آخر بئر نفط، مع إمكانية البحث داخل الأحداث، ومعرفة تاريخ كل حدث، وحفظ الأحداث المفضّلة في قائمة خاصة باستخدام قاعدة البيانات TinyDB.

فكرة المشروع

أولاً: واجهة المصمم Designer

1. إنشاء شاشة البداية (Screen1) تتضمن:

المكونات	وصف المكون	تسمية مقترحة للمكون	ملاحظات
Image	صورة	KwLogo	شعار دولة الكويت
Label	تسمية	AppTitle	عنوان التطبيق: Bright Moments In Kuwaitfis History
Button	زر	BtnStart	الانتقال إلى الشاشة التالية

2. إنشاء الشاشة الرئيسية (MainScreen) تتضمن:

الشاشة الرئيسية MainScreen تستخدم للتنقل بين شاشات التطبيق المختلفة
(Screen1- KwFlag_Color- Kwknowledge- ScreenSound)

المكونات	وصف المكون	تسمية مقترحة للمكون	ملاحظات
Label	تسمية	LblWelcome	كتابة رسالة ترحيبية : Welcome To Our App
Button1	زر	BtnFlag	للانتقال إلى شاشة KwFlag_Color
Button2	زر	BtnHistory	للانتقال إلى شاشة Kwknowledge

Button3	زر	BtnKwAnthem	للانتقال إلى شاشة ScreenSound
Button4	زر	BtnHome	للعودة إلى شاشة البداية Screen1 عند الحاجة
Horizontal Arrangement	الترتيب الأفقي	-	إدراج ثلاثة منها لتنسيق عناصر الشاشة

3. شاشة ألوان علم الكويت KwFlag_Color.

المكونات	وصف المكون	تسمية مقترحة للمكون	ملاحظات
Image	صورة	ImgFlag	عرض أبيات شعر لعلم الكويت باللغة العربية
Label1	تسمية	LblPoem1	أبيات الشعر المستوحاة منها ألوان علم الكويت: إنّا لَقَومٌ أَبَت أَخلاقُنا شَرَفاً
Label2	تسمية	LblPoem2	أبيات الشعر المستوحاة منها ألوان علم الكويت: أَن نبتدئ بِالأَذى مَن لَيسَ يُؤذينا
Label3	تسمية	LblPoem3	أبيات الشعر المستوحاة منها ألوان علم الكويت: بيضٌ صَنائِعُنا سودٌ وَقائِعُنا
Label4	تسمية	LblPoem3	أبيات الشعر المستوحاة منها ألوان علم الكويت: خُضرٌ مَرابِعُنا حُمرٌ مَواضينا
Label4	تسمية	LblColor	إظهار اللون عند الضغط على الزر الخاص باللون
Button1	زر	BtnBlack	تغيير لون LblColor إلى اللون الأسود
Button2	زر	BtnGreen	تغيير لون LblColor إلى اللون الأخضر

Button3	زر	BtnWhite	تغيير لون LblColor إلى اللون الأبيض
Button4	زر	BtnRed	تغيير لون LblColor إلى اللون الأحمر
Button5	زر	Btn_ColorBack	للانتقال إلى الشاشة MainScreen
Horizontal Arrangement	الترتيب الأفقي	-	إدراج 4 منها لتنظيم العناصر

4. شاشة المعرفة التاريخية Kwknowledge.

المكونات	وصف المكون	تسمية مقترحة للمكون	ملاحظات
DatePicker	التاريخ	DatePicker1	عرض التاريخ
Label1	تسمية	LblDate	عرض التاريخ عند فتح الشاشة
Label2	تسمية	LblSearch	عرض نتيجة البحث (اسم الحدث وتاريخه).
Label3	تسمية	LblStatuse	رسالة تأكيد على أنه تم الحفظ في المفضلة.
Label4	تسمية	LblFavorites	عرض قائمة البحث المفضّلة للمستخدم.
TextBox	مربع النص	TextInfo	إدخال اسم الحدث التاريخي للبحث من قبل المستخدم.
Button1	زر	BtnSearch	تنفيذ عملية البحث.
Button2	زر	BtnAddFa-vorites	إضافة الحدث إلى قائمة المفضلات
Button3	زر	BtnBackN	الرجوع إلى شاشة MainScreen
TinyDB	قاعدة بيانات	MainDB	حفظ بيانات البحث المفضلة واسترجاعها.

5. شاشة السلام الوطني ScreenSound.

المكونات	وصف المكون	تسمية مقترحة للمكون	ملاحظات
Title	عنوان الشاشة	-	Kuwaiti National Anthem
Label1	تسمية	LblAnthemTitle	عنوان السلام الوطني
Button1	زر	BtnSound	تشغيل السلام الوطني - KwAnthem
Button2	زر	BtnSoundStop	إيقاف السلام الوطني
Button3	زر	BtnBackN	الرجوع إلى شاشة MainScreen
Horizontal Arrangement	-	-	تنظيم العناصر في الشاشة

خطوات تنفيذ التطبيق

ثانياً: واجهة الكتل البرمجية Blocks

منطق البرمجة في شاشة Kwknowledge.

أولاً: تعريف القوائم والمتغيرات - Lists & Variables.

1. قائمة الأحداث (EventName) .
   - تاريخ الاستقلال.
   - دستور الكويت.
   - تاريخ التحرير.
   - إطفاء آخر بئر بترول.
2. قائمة السنوات (years) .
   - 1961
   - 1962
   - 1991
   - 1991

3. المتغيرات .

- favorites : قائمة المفضلة.
- found: مؤشر العثور على الحدث (صواب/خطأ).
- i : عدّاد الحلقة.
- foundIndex : موضع الحدث في القائمة.

ثانياً: الإجراءات - Procedures

1. إجراء البحث searchAction

- يقرأ نصّ الحدث من TextBox.
- يمرّ على قائمة EventName بحلقة تكرار ويقارن نصيًا (معالجة فراغات/حالة الأحرف إن لزم).
- عند التطابق: يعيّن found = true و foundIndex = i ثم يتوقّف.

2. إجراء العرض displayInfo

- إذا كان found = true: يعرض نتيجة البحث في LblSearch (اسم الحدث مع تاريخه الموافق من years باستخدام foundIndex.
- إذا لم يُعثر عليه: يعرض رسالة مناسبة (مثل: «لم يتم العثور على الحدث»).

3. إجراء عرض المفضلة displayFavorites

- يحوّل عناصر favorites إلى على شكل قائمة نصية.
- عرضها في LblFavorites.

ثالثاً: برمجة الأزرار - Buttons

1. زر البحث BtnSearch

- عند الضغط عليه: يستدعي searchAction ثم displayInfo.

2. زر الإضافة للمفضلة BtnAddFavorite

- الشرط: إذا كان found = true.
- إضافة الحدث (مع السنة إن رغبتِ) إلى favorites.
- حفظ القائمة المحدثة في TinyDB تحت وسم (Tag) مناسب مثل favoritesList.
- استدعاء الإجراء displayFavorites لتحديث واجهة المفضلة.
- عرض رسالة تأكيد في Label مخصص (مثلاً: «تم الحفظ في المفضلة»).

رابعاً : تهيئة الشاشة - Initialize

1. عند فتح شاشة Kwknowledge .

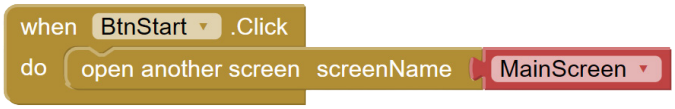
- تحميل قائمة favorites من TinyDB. (إذا كانت فارغة: تُنشأ قائمة جديدة).
- استدعاء الإجراء displayFavorites لعرض القائمة المفضلة السابقة.
- استخدام DatePicker لتعيين التاريخ الحالي في LblDate .

ملحوظات تحسين (اختيارية):

- توحيد المقارنة النصية بتحويل النصين إلى أحرف صغيرة وإزالة الفراغات الجانبية قبل المطابقة.
- عند الحفظ في المفضلة، يُفضّل التحقق من عدم التكرار.
- تسمية الوسم في TinyDB بوضوح مثل : favoritesList لتسهيل الصيانة.

الكتل البرمجية - الشاشة الأولى Screen1:

- عند الضغط على الزر BtnStart.
- ينتقل الى الشاشة MainScreen .



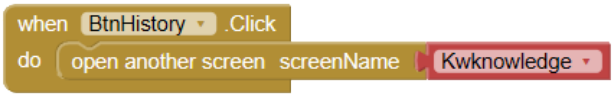
الكتل البرمجية - الشاشة الرئيسية MainScreen:

- تستخدم للتنقل بين شاشات التطبيق المختلفة:
(Screen1- - KwFlag_Color- Kwknowledge- ScreenSound)
- يتم إضافة الكتل البرمجية اللزمة لكل زر للانتقال للشاشة الخاصة له :



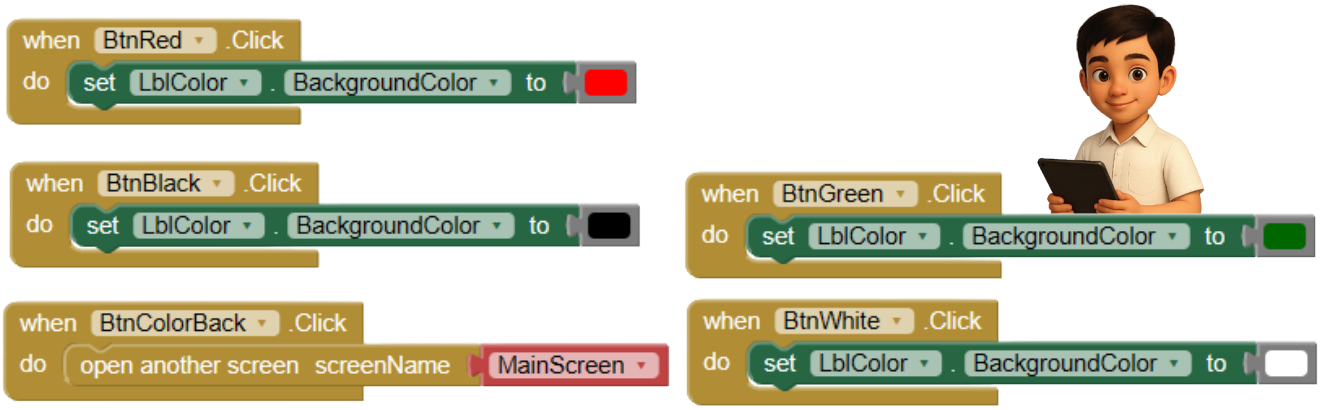






الكتل البرمجية - شاشة KwFlag_Color:

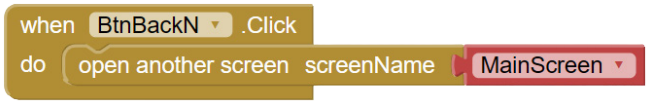
- عرض ألوان علم دولة الكويت ومما هي مستوحاة وما تمثّله من دلالات شعرية ووطنية.
- تغيير لون خلفية (Label) برمجيًا حسب لون الزر الذي يضغطه المستخدم.
- إضافة الكتل البرمجية اللازمة لتغير لون Label مع ضغط الزر حسب اللون .



الكتل البرمجية - شاشة Kwknowledge:

عرض معلومات تاريخية من لحظات تاريخية مضيئة في تاريخ كويتنا الحبيبة :

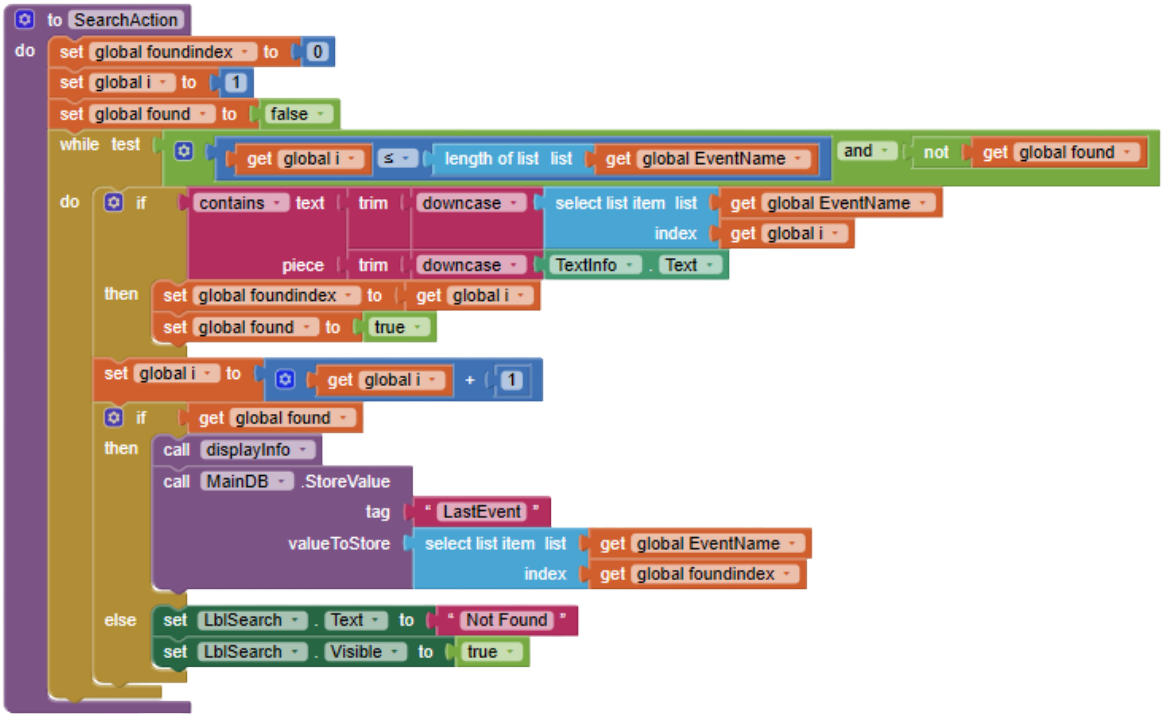
- الحدث : (دستور الكويت - تاريخ الاستقلال - تاريخ التحرير - إطفاء اخر بئر بترول).
- السنة الخاصة بكل حدث: (1961-1962-1991-1991).
  - عند ضغط زر BtnBackN للرجوع الى صفحة MainScreen



- إضافة المتغيرات Variables

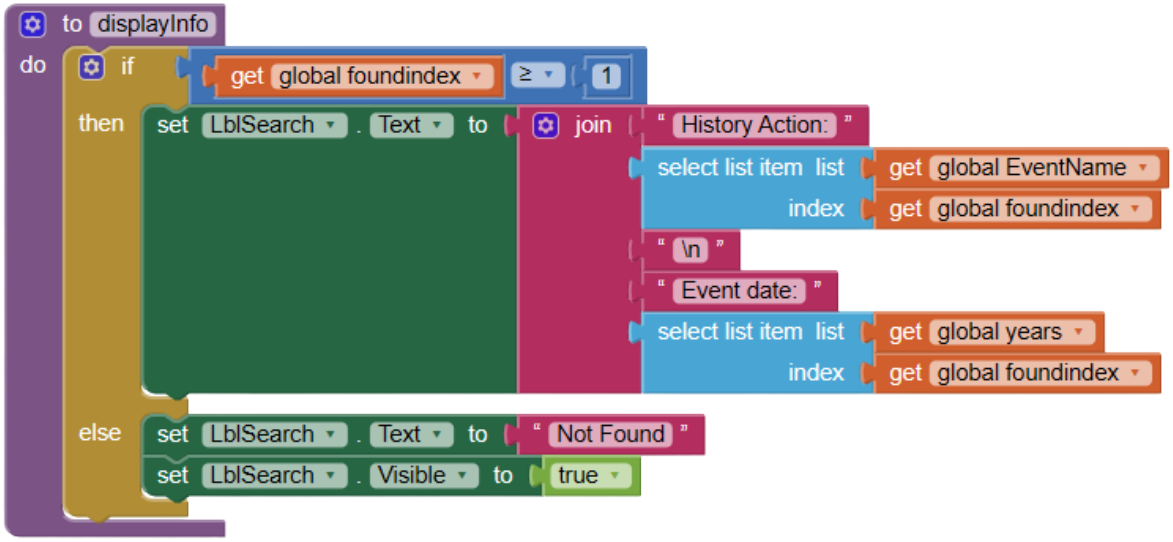


- إنشاء الإجراء Procedures خاص بالبحث: searchAction

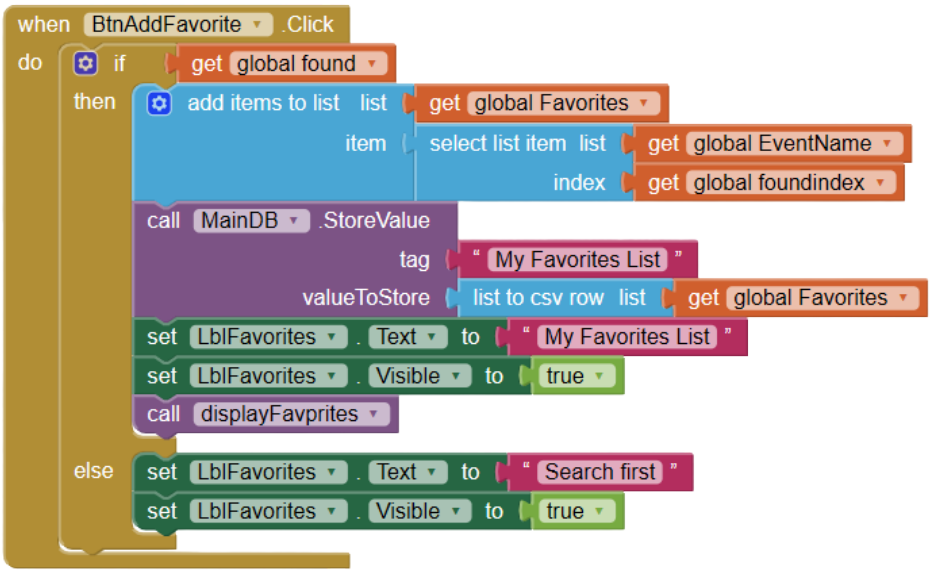




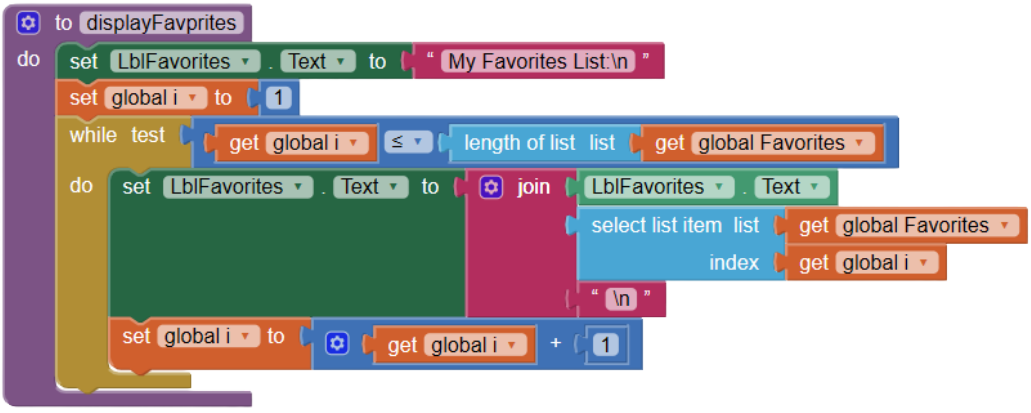
- عند الضغط على الزر BtnSearch يتم تشغيل الإجراء SearchACtion
- إنشاء الإجراء Procedures displayInfo : خاص بالبحث لعرض نتيجة البحث في LblSearch.



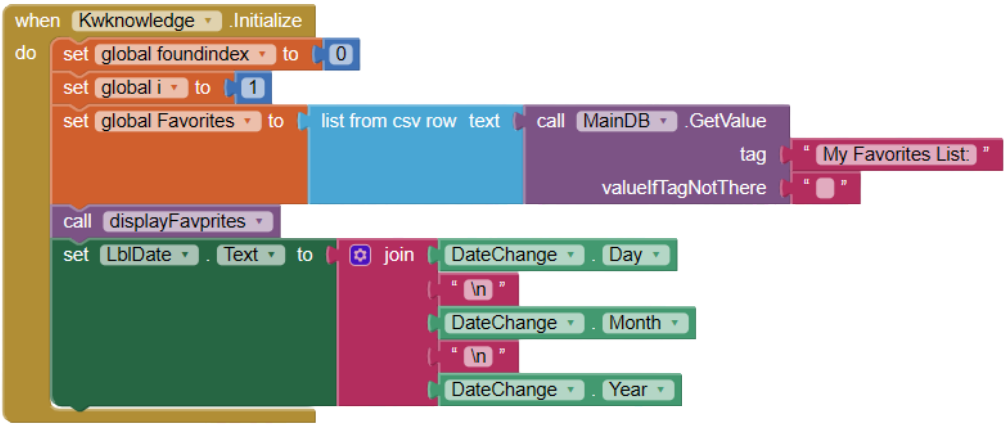
- عند الضغط على الزر BtnAddFavorite يتم إضافة نتيجة البحث للمفضلة.



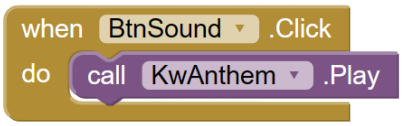
- إنشاء الإجراء Procedures displayFavorites: الخاص بعرض المفضلة لنتيجة البحث في LblStatuse.



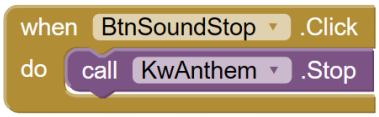
- عند فتح الشاشة Kwknowledge يتم استدعاء الإجراء displayFavorites وإضافة التاريخ داخل LblDate.



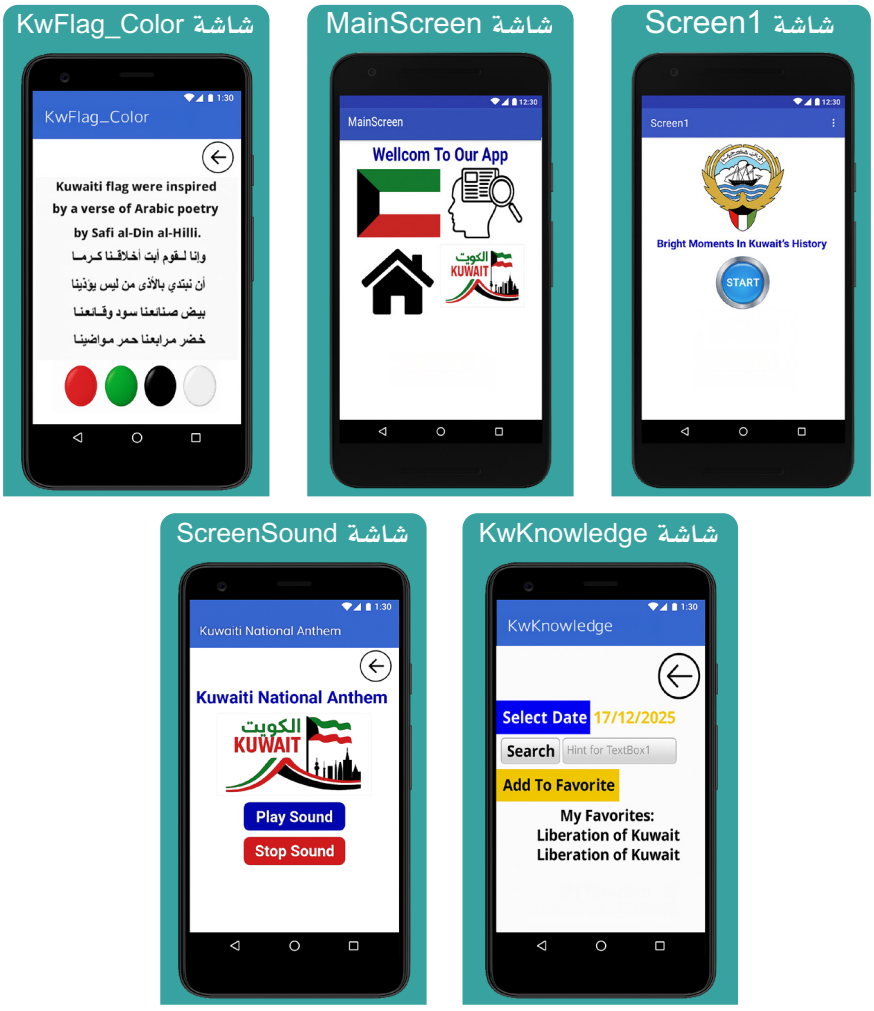
- **الكتل البرمجية - شاشة ScreenSound :**
  - صفحة ScreenSound لعرض السلام الوطني الكويت و زر لإيقافه.
  - عند الضغط على الزر BtnSound يتم عرض السلام الوطني .

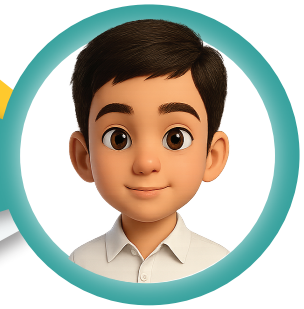


  - عند الضغط على الزر BtnSoundStop يتم إيقاف السلام الوطني .



شكل شاشات التطبيق في المحاكي :





نماذج لأفكار مشروعات متنوعة - تطبيقات App Inventor

مشروعات وطنية وتاريخية

- تصميم تطبيق يعرض معالم وآثار الكويت على خريطة تفاعلية مع صور ووصف قصير وتاريخ الإنشاء.
- تصميم تطبيق قصص تاريخية تفاعلية عن محطات من تاريخ الكويت مع أسئلة قصيرةبعد كل قصة.
- تصميم تطبيق خط زمني (Timeline) للأحداث الوطنية المهمة مع إمكانية البحث عن الحدث بالسنة أو الكلمة المفتاحية.

مشروعات تعليمية للمواد الدراسية

- تصميم تطبيق مراجعة دروس الحاسوب يتضمن بنك أسئلة واختبارات قصيرة مع تصحيح فوري.
- تصميم تطبيق تعلّم مفردات اللغة الإنجليزية (كلمات + صورة + نطق صوتي + اختبار بسيط).
- تصميم تطبيق رياضيات لحل مسائل الجمع والطرح والضرب مع مستويات صعوبة وسجل لنتائج الطالب.

مشروعات مهارات حياتية وصحية

- تصميم تطبيق عادات صحية يتابع شرب الماء، والنوم، والنشاط البدني مع تذكيرات يومية.
- تصميم تطبيق يسجّل مشاعر الطالب اليومية بطريقة مبسّطة باستخدام رموز بسيطة.
- تصميم تطبيق دليل الإسعافات الأولية للأطفال مع خطوات مبسطة وصور توضيحية.

مشروعات خدمية بسيطة

- تصميم تطبيق قائمة مهام (To-Do / Shopping List) لإدارة المهام اليومية مع إمكانية وضع علامة على المهام المنجزة.
- تصميم تطبيق منظم دراسي لتسجيل الواجبات والاختبارات والتنبيه بالمواعيد المهمة.
- تصميم تطبيق دليل مدرستي يتضمن خريطة المدرسة، مواعيد الحصص، أرقام التواصل المهمة.

مشروعات ترفيهية وتفاعلية

- تصميم لعبة أسئلة وأجوبة عن الكويت (ثقافة، تاريخ، جغرافيا) تتضمن نظام نقاط ولوحة شرف لعرض أفضل اللاعبين.
- تصميم تطبيق رسم بسيط (Paint) يسمح بالرسم بالأصبع وتغيير الألوان ومسح اللوحة.
- تصميم لعبة تحدي السرعة مثل الضغط على زر أو تحريك كرة على الشاشة (Ball Bounce) مع وجود عدّاد لحساب النتيجة.



نصائح للمتعلمين أثناء تطبيق المشروع

- **أولاً : تنظيم الوقت وإدارة العمل**
  - تنظيم الوقت ووضع خطة واضحة لتنفيذ المهام.
  - تقسيم المشروع إلى مهام صغيرة مع تحديد وقت لكل مهمة في جدول أو مخطط بسيط.
  - متابعة التقدّم بانتظام (ما تم إنجازه / ما تبقى) وتعديل الخطة عند الحاجة.

- **ثانياً: الفهم وطرح الأسئلة**
  - قراءة المطلوب بدقة وفهم خطوات التنفيذ.
  - تدوين الأسئلة غير الواضحة وطرحها على المعلم في وقت مبكر لتجنب الأخطاء.

- **ثالثاً: العمل الجماعي والتعاون**
  - توزيع الأدوار بين أفراد المجموعة بوضوح (باحث، كاتب، مبرمج، مصمم، عارض).
  - احترام آراء الزملاء والاستماع لوجهات النظر المختلفة.
  - التعاون في مساعدة الأعضاء المتأخرين لضمان توازن العمل داخل الفريق.

- **رابعاً: البحث والدقة العلمية**
  - استخدام مصادر موثوقة ومتنوعة (كتب، مواقع رسمية، منصات تعليمية) وتجنب المعلومات غير الموثوقة.
  - تدوين الملاحظات أثناء البحث وترتيبها في نقاط مختصرة.
  - الربط بين ما يتم تعلمه في المشروع وما تمت دراسته في الكتاب أو الحصص.

- **خامساً: العرض والتقويم الذاتي**
  - تجربة التطبيق أكثر من مرة للتأكد من خلوّه من الأخطاء قبل التسليم.
  - إعداد عرض واضح ومنظّم ثم التدرب عليه قبل تقديمه.
  - استخدام قائمة تحقق للتأكد من استيفاء جميع متطلبات المشروع قبل التسليم.

•David, M. (2014). Developing Android apps using the MIT App Inventor (2nd ed.). ICPM Publications.

•Easttom, C. (2023). Programming with MIT App Inventor: A hands-on approach. Independently published.

•MIT App Inventor Team. (2025). MIT App Inventor MIT. https://appinventor.mit.edu/explore/resources

•MIT Center for Mobile Learning. (n.d.). Teach with App Inventor [Curriculum resources]. MIT App Inventor. Retrieved December 17, 2025, from https://appinventor.mit.edu/explore/teach

•UQ Library. (2019). Software and mobile apps - APA 7th referencing style. https://guides.library.uq.edu.au/referencing/apa7/software-apps

•Wolber, D., Abelson, H., Spertus, E., & Looney, L. (2014). App Inventor 2: Create your own Android apps. O÷Reilly Media.

9