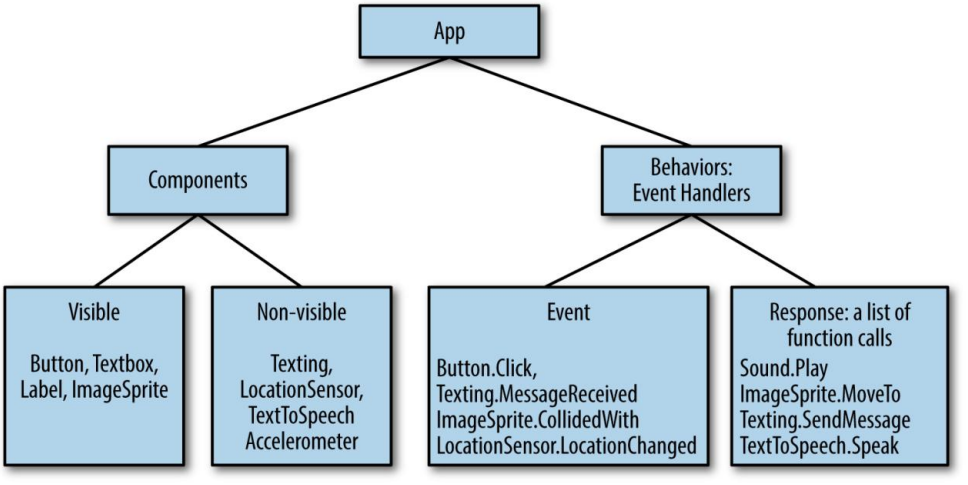
הבנת הארכיטקטורה של אפליקציה

איור .14-1

פרק זה בוחן את המבנה של אפליקציה מנקודת מבטו של מתכנת. זה מתחיל באנלוגיה המסורתית שאפליקציה היא כמו מתכון ואז ממשיך להמשיג מחדש אפליקציה כמערכת של רכיבים המגיבים לאירועים. הפרק גם בוחן כיצד אפליקציות יכולות לשאול שאלות, לחזור, לזכור ולדבר עם האינטרנט, כל אלו יתוארו ביתר פירוט בפרקים מאוחרים יותר.

אנשים רבים יכולים לומר לך ממה היא אפליקציה נקודת מבט של משתמש, אבל להבין מה זה מנקודת מבט של מתכנת היא מסובכת יותר. לאפליקציות יש מבנה פנימי שעליך להבין כדי ליצור אותן בצורה יעילה.

אחת הדרכים לתאר את החלקים הפנימיים של האפליקציה היא לחלק אותה לשני חלקים, מרכיביה והתנהגויותיה. בקירוב, אלה תואמים לשני החלונות העיקריים שבהם אתה משתמש באתה משתמש Component Designer-בכדי לציין את האובייקטים (הרכיבים) של האפליקציה, ואתה משתמש Blocks Editor-בכדי לתכנת כיצד האפליקציה מגיבה למשתמש ולאירועים חיצוניים (התנהגות האפליקציה).
איור 14-1מספק סקירה כללית של ארכיטקטורת אפליקציה זו. בפרק זה, נעשה זאת לחקור את הארכיטקטורה הזו בפירוט.



איור .14-2הארכיטקטורה הפנימית של אפליקציית App Inventor

רכיבים

ישנם שני סוגים עיקריים של רכיבים באפליקציה: גלוי ולא גלוי. הרכיבים הגלויים של האפליקציה הם אלה שתוכל לראות כאשר האפליקציה מופעלת -לחצנים, תיבות טקסט ותוויות. אלה מכונים לעתים קרובות ממשק המשתמש של האפליקציה.

רכיבים שאינם גלויים הם אלו שאינך יכול לראות, כך שהם אינם חלק מהמשתמש מִמְשָׁק. במקום זאת, הם מספקים גישה לפונקציונליות המובנית של המכשיר; לדוגמה, רכיב ה-SMS שולח ומעבד טקסטים של ,SMSרכיב ה- LocationSensorקובע את מיקום המכשיר, ורכיב TextToSpeechמדבר.

הרכיבים הלא גלויים הם הטכנולוגיה בתוך המכשיר -דבורים קטנות שעושות עבודות עבור האפליקציה שלך.

גם רכיבים גלויים וגם לא גלויים מוגדרים על ידי קבוצה של מאפיינים.

מאפיינים הם חריצי זיכרון לאחסון מידע על הרכיב. לרכיבים גלויים כמו לחצנים ותוויות יש מאפיינים כגון Width, Height- ו, Alignmentאשר יחד מגדירים כיצד הרכיב נראה.

מאפייני הרכיב הם כמו תאי גיליון אלקטרוני: אתה משנה אותם Component Designer-ב כדי להגדיר את המראה הראשוני של רכיב. אתה יכול גם לשנות את הערכים עם בלוקים.

התנהגות

רכיבי האפליקציה הם בדרך כלל פשוטים וקלים להבנה: תיבת טקסט מיועדת להזנת מידע, כפתור מיועד ללחיצה וכן הלאה. ההתנהגות של אפליקציה, לעומת זאת, היא קשה מבחינה רעיונית ולעתים קרובות מורכבת. ההתנהגות מגדירה כיצד

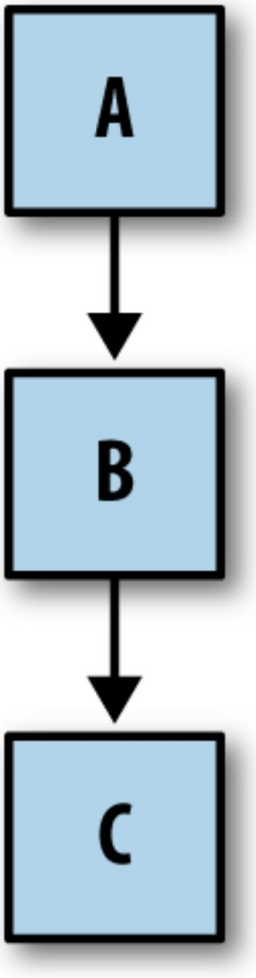
האפליקציה צריכה להגיב לאירועים, הן ביוזמת המשתמש (למשל, לחיצת כפתור) והן חיצונית (למשל, הודעת SMS שמגיעה לטלפון). הקושי לציין התנהגות אינטראקטיבית כזו הוא הסיבה לכך שתכנות הוא כל כך מאתגר.

למרבה המזל, App Inventor מספק שפה מבוססת בלוקים ברמה גבוהה לציון התנהגויות. הבלוקים הופכים התנהגויות תכנות ליותר כמו חיבור חלקי פאזל יחד, בניגוד לשפות תכנות מסורתיות מבוססות טקסט, הכוללות למידה והקלדה של כמויות קוד גדולות. Inventor App-ונועד להקל במיוחד על ציון התנהגויות של תגובה לאירועים. הסעיפים הבאים מספקים מודל להבנת התנהגות האפליקציה וכיצד לציין אותה ב-App Inventor.

אפליקציה כמתכון

באופן מסורתי, תוכנה הושוותה לעתים קרובות למתכון. כמו מתכון, אפליקציה מסורתית עוקבת אחר רצף ליניארי של הוראות שעל המחשב לבצע, כמו למשל באיור 14-2.

אפליקציה טיפוסית עשויה להתחיל עסקה בנקאית (A), לבצע כמה חישובים ו שנה חשבון של לקוח (B), ולאחר מכן הדפס את היתרה החדשה על המסך (C).

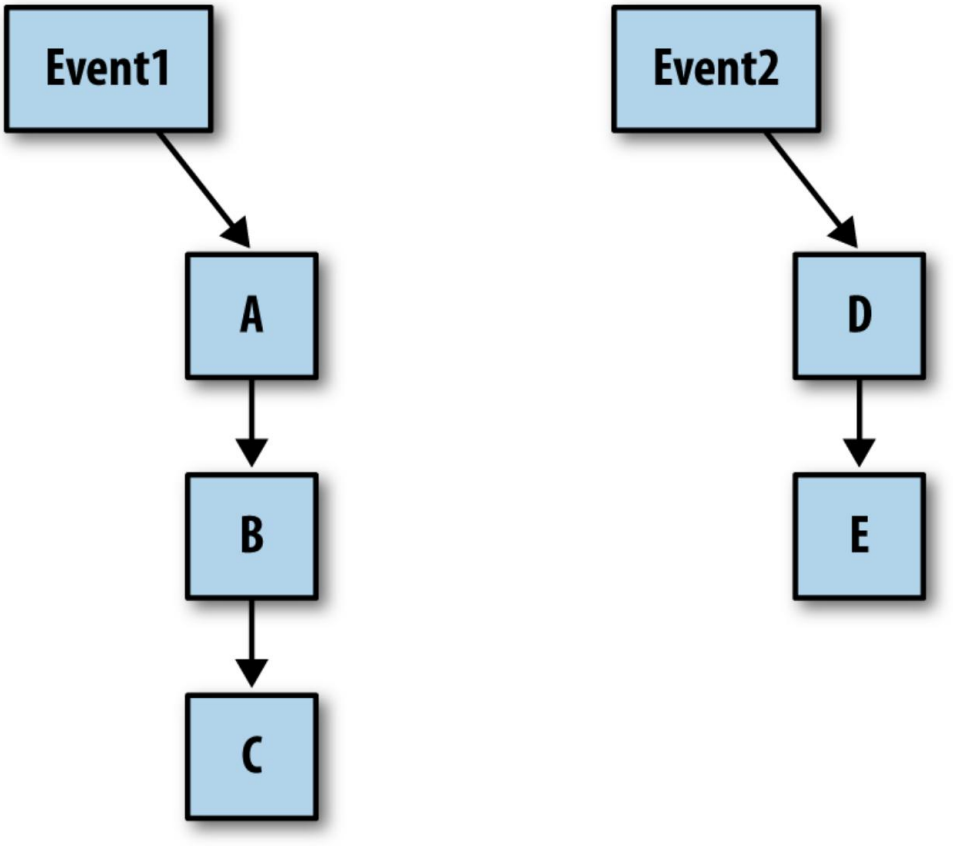


איור .14-3תוכנה מסורתית עוקבת אחר רצף ליניארי של הוראות

אפליקציה כקבוצה של מטפלי אירועים

האפליקציה כפרדיגמת מתכונים מתאימה היטב למחשב מוחץ המספרים המוקדם, אבל היא לא מתאימה לטלפונים ניידים, לאינטרנט ובכלל לרוב המחשוב שנעשה היום. רוב התוכנות המודרניות לא מבצעות שלל הוראות בסדר קבוע מראש; במקום זאת, הוא מגיב לאירועים - לרוב, אירועים שיזמו משתמש הקצה של האפליקציה. לדוגמה, אם המשתמש מקיש על כפתור, האפליקציה מגיבה בביצוע פעולה כלשהי (למשל, שליחת הודעת טקסט). עבור טלפונים ומכשירים עם מסך מגע, פעולת גרירת Fnger-השלך על פני המסך היא אירוע נוסף. ה

האפליקציה עשויה להגיב לאותו אירוע על ידי ציור קו מהנקודה שבה הפנר שלך יוצר קשר ראשון עם המסך ועד לנקודה שבה הרמת אותו.
אפליקציות מודרניות מומשגות טוב יותר כמכונות תגובה לאירועים. האפליקציות כן כולל מתכונים -רצפים של הוראות -אך כל מתכון מבוצע רק בתגובה לאירוע כלשהו, כפי שמוצג באיור .14-3

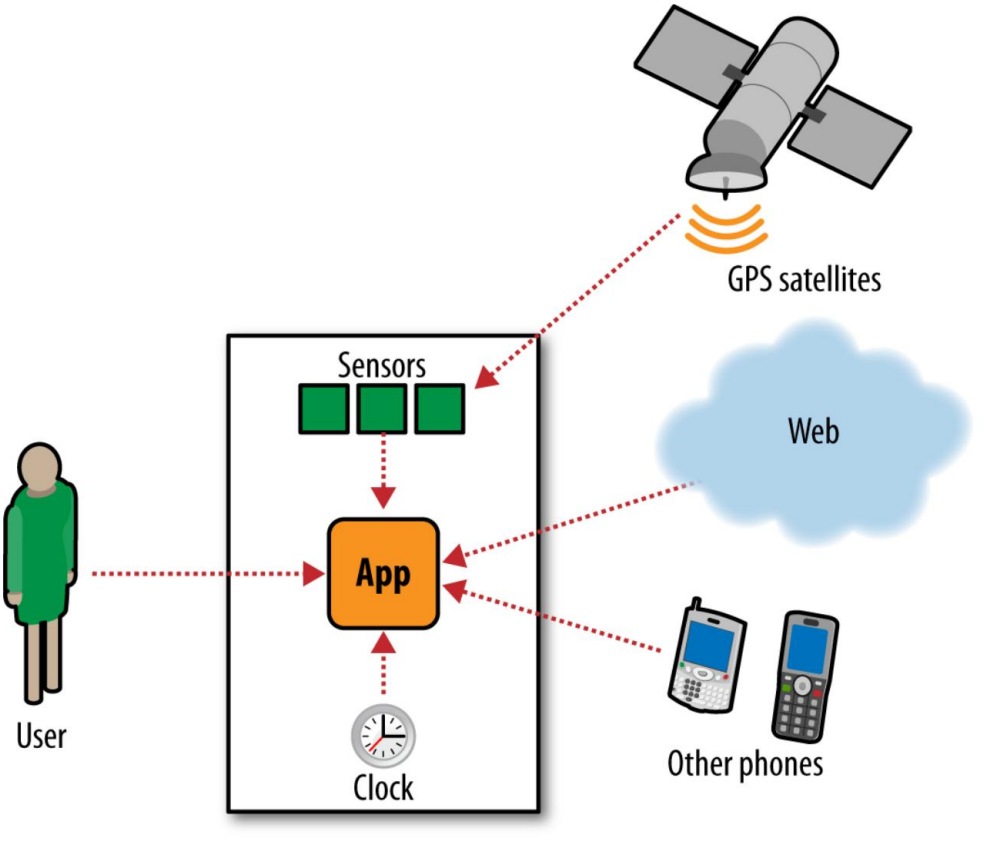


איור .14-4אפליקציה כמספר מתכונים המחוברים לאירועים

כאשר מתרחשים אירועים, האפליקציה מגיבה על ידי קריאה לרצף של פונקציות. פונקציות הן דברים שאתה יכול לעשות עם, או עם, רכיב; אלה יכולות להיות פעולות כמו שליחת טקסט SMS, או פעולות לשינוי מאפיינים כמו שינוי הטקסט בתווית של ממשק המשתמש. להתקשר או להפעיל פונקציה פירושו לבצע את הפונקציה -לגרום לזה לקרות. אנו קוראים לאירוע ולקבוצת הפונקציות שבוצעו בתגובה לו מטפל באירועים.

אירועים רבים יוזמים על ידי משתמש הקצה, אך חלקם לא. אפליקציה יכולה להגיב לאירועים שקורים בטלפון, כגון שינויים בחיישן הכיוון שלה ובשעון (כלומר, הזמן החולף), או שהיא יכולה להגיב לאירועים שמקורם בחוץ

הטלפון, כגון הודעת טקסט או שיחה נכנסת מטלפון אחר, או נתונים המגיעים מהאינטרנט (ראה איור .(14-4



איור .14-5אפליקציה יכולה להגיב הן לאירועים פנימיים והן לאירועים חיצוניים

אחת הסיבות לכך שתכנות App Inventorהוא כל כך אינטואיטיבי היא שהיא מבוססת ישירות על פרדיגמת אירוע-תגובה זו; מטפלי אירועים הם פרימיטיביים בשפה (בהרבה שפות, זה לא המקרה). אתה מתחיל להגדיר התנהגות על ידי גרירת בלוק אירוע, שיש לו את הצורה, "do. "When <event>לדוגמה, שקול אפליקציה, ,SpeakItהמגיבה ללחיצות על כפתורים על ידי דיבור בקול של הטקסט שהמשתמש הקליד בתיבת טקסט. ניתן לתכנת יישום זה עם מטפל באירועים בודדים, כפי שמוצג באיור .14-5

איור 14-6. מטפל באירועים עבור אפליקציית SpeakIt

בלוקים אלו מציינים שכאשר המשתמש לוחץ על הכפתור בשם SpeakItButton, רכיב TextToSpeech צריך לומר את המילים שהמשתמש הקליד בתיבת הטקסט בשם TextBox1. התגובה היא הקריאה לפונקציה TextToSpeech1.Speak. האירוע הוא SpeakItButton.Click. המטפל באירועים כולל את כל הבלוקים באיור 14-5.

עם App Inventor, כל הפעילות מתרחשת בתגובה לאירוע. האפליקציה שלך לא צריכה מכילים בלוקים מחוץ לאירוע של אירוע מתי כן לחסום. לדוגמה, הבלוקים באיור 14-6 לא משיגים דבר כשהם מוצפים לבד.

איור 14-7. בלוקים צפים לא יעשו שום דבר מחוץ למטפל באירועים

סוגי אירועים

האירועים שיכולים לעורר פעילות נכנסים לקטגוריות המפורטות בטבלה 14-1.

טבלה 14-1. אירועים שיכולים לעורר פעילות

קטגוריה	אירוע
אירוע ביוזמת המשתמש	כאשר המשתמש לוחץ על כפתור1, בצע...
אירוע אתחול	כאשר האפליקציה מופעלת, בצע...
אירוע טיימר	כל 20 מילישניות, עשה...
אירוע אנימציה	כאשר שני עצמים מתנגשים, עשה...
אירוע חיצוני	כאשר הטלפון מקבל הודעת טקסט, בצע...

אירועים ביוזמת המשתמש

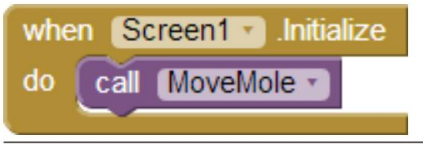
אירועים ביוזמת המשתמש הם סוג האירוע הנפוץ ביותר. עם טפסי קלט, זה בדרך כלל המשתמש שמקיש על כפתור שמפעיל תגובה מהאפליקציה. אפליקציות גרפיות נוספות מגיבות לנגיעות ולגרירות.

אירועי אתחול

לפעמים, האפליקציה שלך צריכה לבצע פונקציות מסוימות מיד עם ההפעלה, לא בתגובה לכל פעילות משתמש קצה או אירוע אחר. איך זה נכנס לפרדיגמת הטיפול באירועים?

שפות לטיפול באירועים כמו App Inventorמחשיבות את השקת האפליקציה כאירוע. אם אתה רוצה שפונקציות ספציפיות יבוצעו עם פתיחת האפליקציה, אתה גורר החוצה בלוק מסך .1אתחול האירוע ומציב בתוכו את בלוקי קריאת הפונקציות הרלוונטיים.

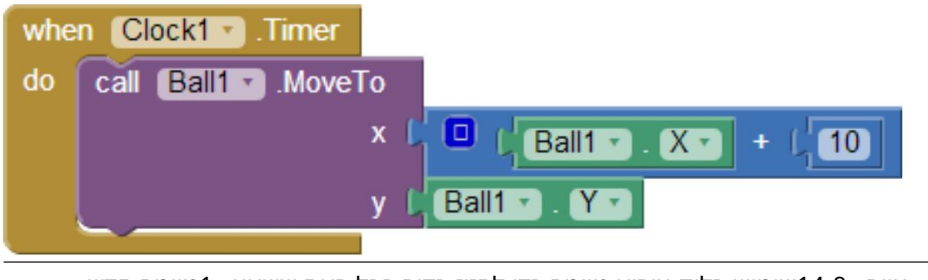
לדוגמה, במשחק MoleMash(פרק 3), הליך MoveMoleנקרא עם הפעלת האפליקציה (ראה איור 14-7כדי למקם את השומה באופן אקראי.



איור 14-8.שימוש בחסום מסך 1.אתחול האירועים כדי להזיז את השומה כאשר האפליקציה מופעלת

אירועי טיימר

פעילות מסוימת באפליקציה מופעלת על ידי הזמן החולף. אתה יכול לחשוב על אנימציה כאובייקט שזז כשהוא מופעל על ידי אירוע טיימר. ל-ppA Inventorיש רכיב שעון שבו אתה יכול להשתמש כדי להפעיל אירועי טיימר. לדוגמה, אם אתה רוצה שכדור על המסך יזוז 10 פיקסלים אופקית במרווח זמן מוגדר, הבלוקים שלך ייראו כמו איור 14-8.



איור 14-9.שימוש בלוק אירוע טיימר כדי להזיז כדור בכל פעם ששעון 1.טיימר חדש

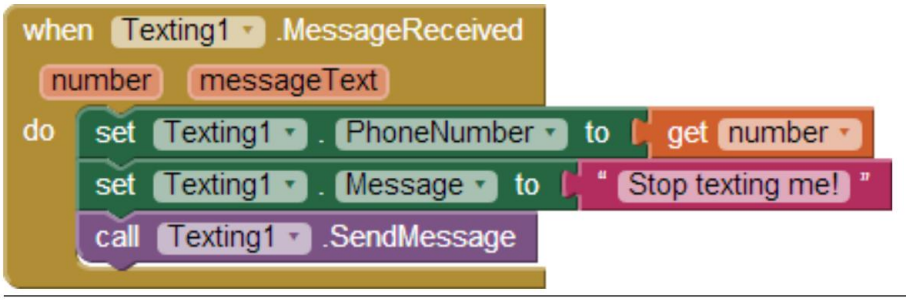
אירועי אנימציה

פעילות הכוללת אובייקטים גרפיים (ספרייטים) בתוך קנבסים תפעיל אירועים. אז אתה יכול לתכנת משחקים ואנימציות אינטראקטיביות אחרות על ידי ציון מה אמור להתרחש באירועים כגון אובייקט שמגיע לקצה הבד או שני אובייקטים מתנגשים, כפי שמתואר באיור 14-9.למידע נוסף, ראה פרק 17.

איור 14-10. כשהספרייט Flying Saucer פוגע באובייקט אחר, השמעת צליל

אירועים חיצוניים

כאשר הטלפון שלך מקבל מידע על מיקום מלווייני GPS, מופעל אירוע. באופן דומה, כאשר הטלפון שלך מקבל הודעת טקסט, מופעל אירוע (איור 14-10).



איור 14-11. האירוע Texting1.MessageReceived מופעל בכל פעם שמתקבל טקסט

כניסות חיצוניות כאלה למכשיר נחשבות לאירועים, לא שונים מהלחץ של המשתמש על כפתור.

לפיכך, כל אפליקציה שתיצור תהיה קבוצה של מטפלי אירועים: אחד לאתחול דברים, חלק שיגיב לקלט של משתמש הקצה, חלק יופעל על ידי זמן, וחלק מופעל על ידי אירועים חיצוניים. התפקיד שלך הוא להמשיג את האפליקציה שלך בצורה זו ולאחר מכן לעצב את התגובה לכל מטפל באירועים.

מטפלי אירועים יכולים לשאול שאלות

התגובות לאירועים אינן תמיד מתכונים ליניאריים; הם יכולים לשאול שאלות ולחזור על פעולות. "שאילת שאלות" פירושה שאילתה לנתונים שהאפליקציה מאחסנת ולקבוע את מהלך (הענף) שלה על סמך התשובות. אנחנו אומרים שלאפליקציות כאלה יש סניפים מותנים. איור 14-11 ממחיש בדיוק ענף כזה.

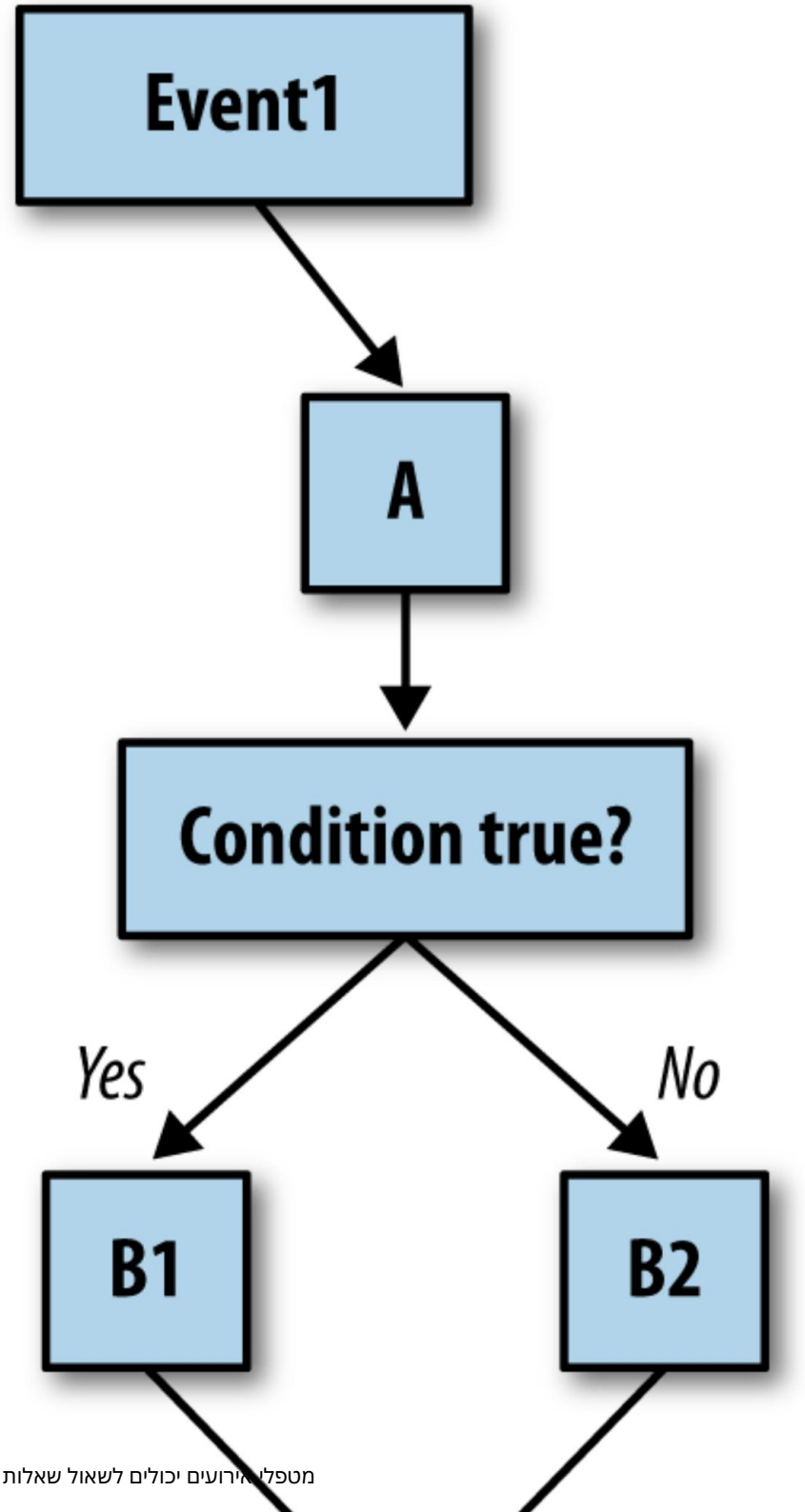
בתרשים, כאשר האירוע מתרחש, האפליקציה מבצעת פעולה A ולאחר מכן בודקת מצב. פונקציה B1 מבוצעת אם התנאי נכון. אם התנאי הוא

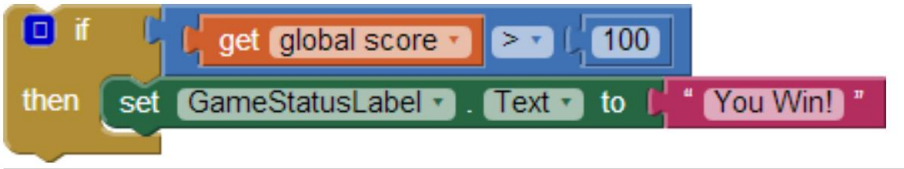
,falseהאפליקציה במקום זאת מבצעת .B2בכל מקרה, האפליקציה ממשיכה לבצע פונקציה C.

מבחנים מותנים הם שאלות כגון "האם הציון הגיע ל-001?" או "עשה את הודעת טקסט שקיבלתי זה עתה הגיעה מג'ו?" מבחנים יכולים להיות גם נוסחאות מורכבות יותר, כולל אופרטורים יחסיים מרובים (קטן מ, גדול מ, שווה ל) ואופרטורים לוגיים (ו, או, לא).

אתה מציין התנהגויות מותנות App Inventor-בבאמצעות החסימות 'אם ואם אחרת '. לדוגמה, הבלוק באיור 12-14ידווח "אתה מנצח!" אם השחקן קלע 100נקודות.

בלוקים מותנים נדונים בפירוט בפרק .18

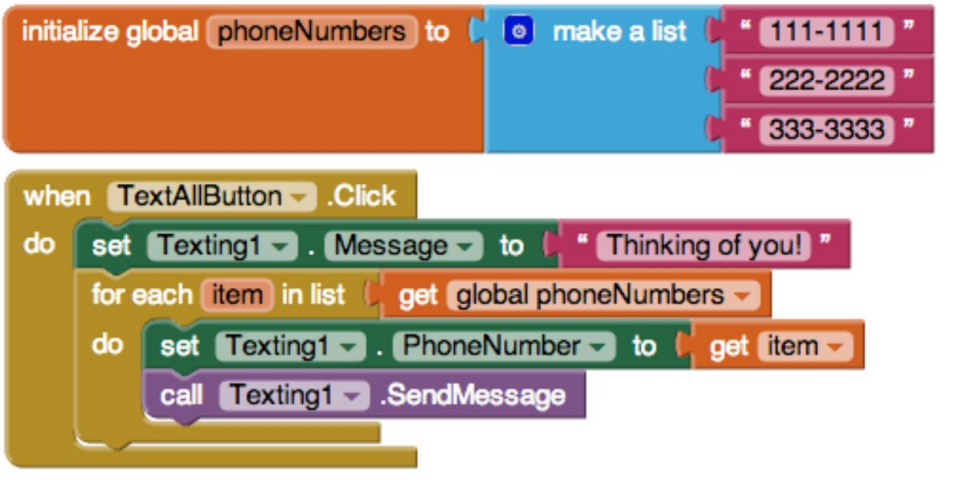




איור 14-13. שימוש בחסימה אם כדי לדווח על ניצחון כאשר השחקן מגיע ל-001 נקודות

מטפלי אירועים יכולים לחזור על חסימות

בנוסף לשאילת שאלות ולהסתעפות על סמך התשובה, תגובה לאירוע יכולה גם לחזור על פעולות מספר פעמים. App Inventorמספק מספר בלוקים לחזרה, כולל ה- while do. For each andשניהם סוגרים בלוקים אחרים. כל הבלוקים בתוך עבור כל אחד מבוצעים פעם אחת עבור כל פריט ברשימה. לדוגמה, אם תרצה לשלוח את אותה הודעה לרשימה של מספרי טלפון, תוכל להשתמש בבלוקים באיור 14-13.



איור 14-14. הבלוקים בתוך עבור כל בלוק חוזרים על עצמם עבור כל פריט ברשימה PhoneNumbers

החסימות בתוך עבור כל בלוק חוזרים על עצמם - במקרה זה, שלוש פעמים, כי הרשימה PhoneNumbersכוללת שלושה פריטים. בדוגמה זו, ההודעה "חושב עליך..." נשלחת לכל שלושת המספרים. בלוקים חוזרים נדונים בפירוט בפרק 20.

מטפלי אירועים יכולים לזכור דברים

מכיוון שמטפל באירועים מבצע חסימות, הוא צריך לעתים קרובות לעקוב אחר מידע. ניתן לאחסן מידע בחריצי זיכרון הנקראים משתנים, אותם אתה מגדיר בעורך הבלוקים. משתנים הם כמו מאפיינים של רכיבים, אבל הם לא משויכים לשום רכיב מסוים. באפליקציית משחק, למשל, אתה יכול להגדיר משתנה שנקרא ניקוד, ומטפלי האירועים שלך ישנו את הערך שלו כשהמשתמש עושה משהו בהתאם. משתנים מאחסנים נתונים באופן זמני בזמן שאפליקציה פועלת; כאשר אתה סוגר את האפליקציה, הנתונים אובדים ואינם זמינים יותר.

לפעמים, האפליקציה שלך צריכה לזכור דברים לא רק בזמן שהיא פועלת, אלא גם כשהיא סגורה ואז נפתחת מחדש. אם עקבת אחר ציון גבוה בהיסטוריה של משחק, למשל, תצטרך לאחסן את הנתונים האלה כך שהם יהיו זמינים בפעם הבאה שמישהו ישחק במשחק. נתונים שנשמרים גם לאחר סגירת אפליקציה נקראים נתונים מתמידים, והם מאוחסנים בסוג כלשהו של מסד נתונים.

נחקור את השימוש הן בזיכרון לטווח קצר (משתנים) והן בזיכרון לטווח ארוך זיכרון (נתוני מסד נתונים) בפרק 16ובפרק ,22בהתאמה.

מטפלי אירועים יכולים ליצור אינטראקציה עם האינטרנט

אפליקציות מסוימות משתמשות רק במידע שבטלפון או במכשיר. אבל אפליקציות רבות מתקשרות עם האינטרנט, או על ידי הצגת דף אינטרנט בתוך האפליקציה, או על ידי שליחת בקשות לממשקי APIשל שירותי אינטרנט (ממשקי תכנות יישומים). אומרים שאפליקציות כאלה הן "מותאמות לאינטרנט".

טוויטר היא דוגמה לשירות אינטרנט שאיתו אפליקציית App Inventorיכולה לדבר. אתה יכול לכתוב אפליקציות שמבקשות ומציגות ציוצים קודמים של החברים שלך וגם לעדכן את סטטוס הטוויטר שלך. אפליקציות שמדברות עם יותר משירות אינטרנט אחד נקראות .mashups נסקור אפליקציות התומכות באינטרנט בפרק .24

סיכום

יוצר אפליקציה חייב לראות את האפליקציה שלו הן מנקודת מבט של משתמש הקצה והן מנקודת מבטו של מתכנת. עם ,App Inventorאתה מעצב איך אפליקציה נראית ואתה מעצב את ההתנהגות שלה -קבוצת המטפלים באירועים שגורמים לאפליקציה להתנהג כפי שאתה רוצה. אתה בונה את מטפלי האירועים האלה על ידי הרכבה והגדרת בלוקים המייצגים אירועים, פונקציות, ענפים מותנים, לולאות חוזרות, שיחות אינטרנט, פעולות מסד נתונים ועוד, ואז בודקים את העבודה שלך על ידי הפעלת האפליקציה בטלפון שלך. לאחר שכותבים כמה תוכניות, מתברר המיפוי בין המבנה הפנימי של אפליקציה לביטוי הפיזי שלה. כשזה קורה, אתה מתכנת!