הגדרת נהלים ושימוש חוזר בלוקים

איור 21-1.

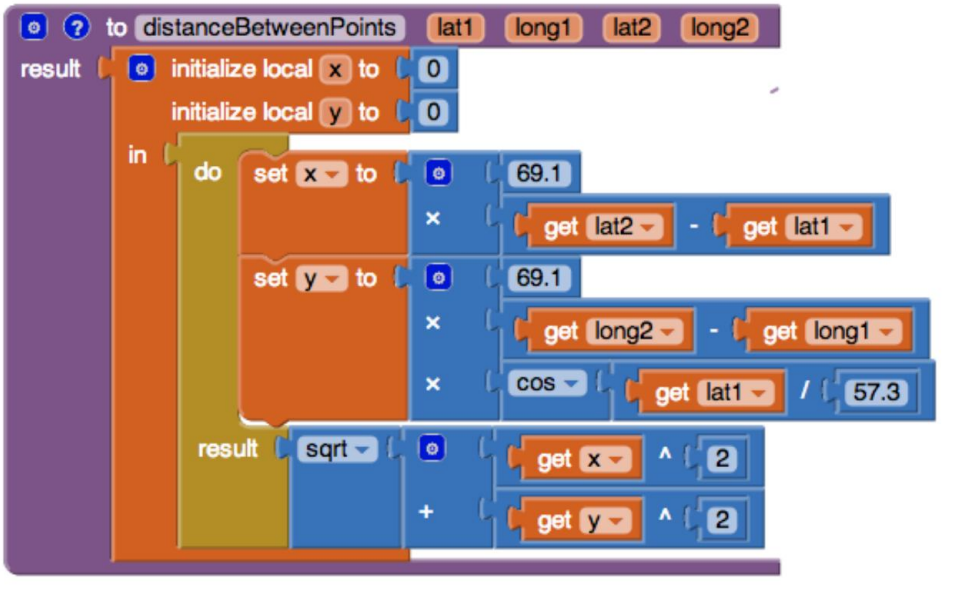
שפות תכנות כגון App Inventorמספקות סט בסיס של פונקציונליות מובנית -במקרה של App Inventor, סט בסיס של בלוקים. שפות תכנות מספקות גם דרך להרחיב את הפונקציונליות על ידי הוספת פונקציות חדשות (בלוקים) לשפה. ב-ppA Inventor, אתה עושה זאת על ידי הגדרת נהלים - המכונים רצפים של בלוקים -שהאפליקציה שלך יכולה לקרוא להם בדיוק כפי שהיא קוראת לבלוקים המוגדרים מראש של App Inventor. כפי שתראה בפרק זה, היכולת ליצור הפשטות כאלה חשובה מאוד לפתרון בעיות מורכבות, שהיא אבן היסוד של בניית אפליקציות מושכות באמת.

כשהורים אומרים לילדם, "לך לצחצח שיניים לפני השינה", הם באמת מתכוונים, "קח את מברשת השיניים ומשחת השיניים שלך מהארון, סחט קצת משחת שיניים על המברשת, סובב את המברשת על כל שן למשך 10שניות (חה!) ," וכולי. "צחצח שיניים" הוא הפשטה: שם מוכר לרצף של הוראות ברמה נמוכה יותר. במקרה זה, ההורים מבקשים מהילד לבצע את ההוראות שכולם הסכימו שמשמעותן "לצחצח שיניים".

בתכנות, אתה יכול ליצור רצפים בעלי שם כאלה של הוראות. שפות תכנות מסוימות מכונות אותן פונקציות או תת-תוכניות. ב-ppA Inventor הם נקראים פרוצדורות. פרוצדורה היא רצף שם של בלוקים שניתן להתקשר אליהם מכל מקום באפליקציה.

איור 21-11הוא דוגמה לנוהל שמעריך את המרחק, במיילים, בין שתי קואורדינטות GPSשאתה שולח אליו.

1בלוקים אלה מוצגים עם כניסות מוטבעות, מה שמקטין את רוחב הבלוקים. אתה יכול ללחוץ לחיצה ימנית על בלוקים כדי לעבור בין קלט "מוטבע" ו"חיצוני".



איור 21-2. נוהל לחישוב המרחק בין נקודות

אל תדאג יותר מדי לגבי הפנימיות של הליך זה עדיין; כל מה שאתה צריך
מבין כרגע שהנהלים כמו זה מאפשרים לך להרחיב את השפה שבה אתה מעצב ובונה
תוכניות. אם כל הורה היה צריך להסביר את הצעדים של "צחצח שיניים" לילד שלו או שלה
בכל לילה, ייתכן שהילד הזה לא יגיע לכיתה ה'. זה הרבה יותר יעיל פשוט להגיד "צחצח
שיניים", וכולם יכולים להמשיך ולהגיע למיטה בשעה סבירה.

באופן דומה, לאחר שתגדיר את הפרוצדורה מרחק בין נקודות, תוכל להתעלם
את הפרטים של איך זה עובד ופשוט עיין (קרא) בשם הפרוצדורה בעת עיצוב או קידוד
של אפליקציה גדולה יותר. סוג זה של הפשטה הוא המפתח לפתרון בעיות גדולות
ומאפשר לנו לפרק פרויקט תוכנה גדול לנתחי קוד ניתנים לניהול.

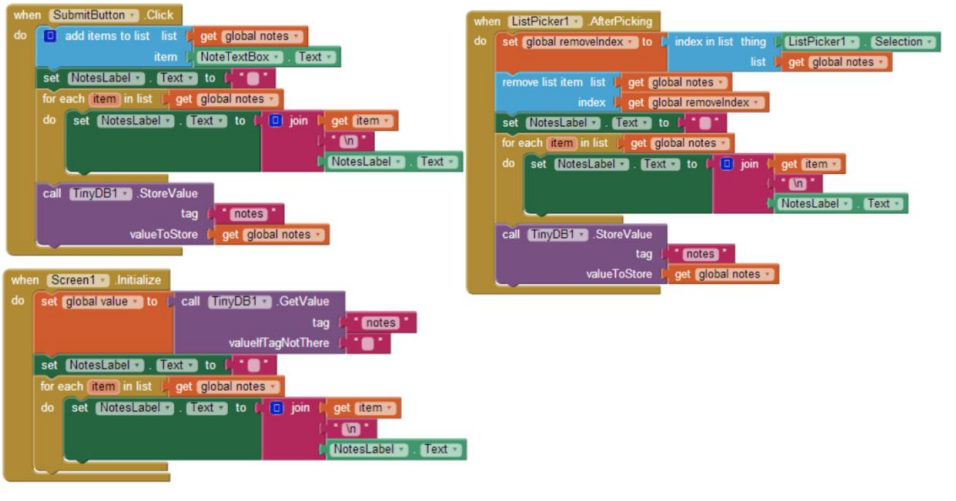
נהלים גם עוזרים להפחית שגיאות מכיוון שהם מבטלים יתירות בקוד שלך. בעזרת נהלים,
אתה יכול לשים גוש קוד במקום אחד ואז לקרוא לו ממקומות שונים ברחבי האפליקציה שלך.
לכן, אם אתה בונה אפליקציה שצריכה לדעת את המרחק המינימלי בין המיקום הנוכחי שלך
ל-01 נקודות אחרות, אין לך צורך ב-01 עותקים של הבלוקים המוצגים באיור 21-1.במקום
זאת, אתה פשוט מגדיר את הליך ה- distanceBetweenPointsואז קורא לו בכל פעם
שאתה צריך. האלטרנטיבה -העתקה והדבקה של בלוקים -תלויה בקוד הרבה יותר (זכור את
הדיון מפרק 19)וכתוצאה מכך, מועדת לשגיאות, כי כאשר אתה מבצע שינוי, אתה צריך
למצוא את כל שאר העותקים של אותם בלוקים ולשנות כל אחד מהם. אחד באותו אופן. תאר
לעצמך שאתה מנסה למצוא את 5עד 10המקומות שבהם הדבקת א

גוש קוד מסוים באפליקציה עם 1,000שורות או בלוקים! פרוצדורה מאפשרת לך במקום זאת לכלול בלוקים במקום אחד, ואז לקרוא לזה פעמים רבות.

נהלים גם עוזרים לך לבנות ספריית קוד שניתן לעשות בה שימוש חוזר ברבים אפליקציות. גם כאשר בונים אפליקציה למטרה מאוד ספציפית, מתכנתים מנוסים תמיד חושבים על דרכים ליצור את הקוד בצורה כזו שתוכל לעשות בו שימוש חוזר באפליקציות אחרות. חלק מהמתכנתים אף פעם לא יוצרים אפליקציות, אלא מתמקדים אך ורק בבניית ספריות קוד ניתנות לשימוש חוזר עבור מתכנתים אחרים לשימוש באפליקציות שלהם!

ביטול יתירות

הבלוקים באיור 2-21הם מאפליקציית Note Taker. תסתכל על הבלוקים וראה אם אתה יכול לזהות את המיותרים.



איור 21-3. אפליקציית הערות עם קוד מיותר

הבלוקים המיותרים הם אלה הכוללים aעבור כל בלוק (למעשה עבור כל אחד והבלוקים המקוננים שלו והסט NotesLabel.Textשמעליו). בשלושתם עבור כל מקרים, תפקידו של הבלוק הוא להציג את רשימת ההערות. באפליקציה זו, התנהגות זו צריכה להתרחש בשלושה מטפלי אירועים: כאשר פריט חדש מתווסף, כאשר פריט מוסר, וכאשר הרשימה נטענת ממסד הנתונים עם השקת האפליקציה.

כשמתכנתים מנוסים רואים יתירות כזו, פעמון נדלק בראשם, כנראה עוד לפני שהם העתיקו והדביקו את הבלוקים מלכתחילה.

הם יודעים שעדיף לכלול יתירות כזו בהליך, גם כדי

להפוך את התוכנית למובנת יותר וכך יהיה הרבה יותר קל לבצע שינויים מאוחר יותר.

בהתאם לכך, מתכנת מנוסה ייצור נוהל, יעביר עותק
של הבלוקים המיותרים לתוכו, ולאחר מכן קרא להליך משלושת המקומות המכילים את
הבלוקים המיותרים. האפליקציה לא תתנהג בצורה שונה, אבל היא תהיה קלה יותר לתחזוקה
וקל יותר למתכנתים אחרים לעבוד איתם. ארגון מחדש של קוד (בלוק) כזה נקרא refactoring.

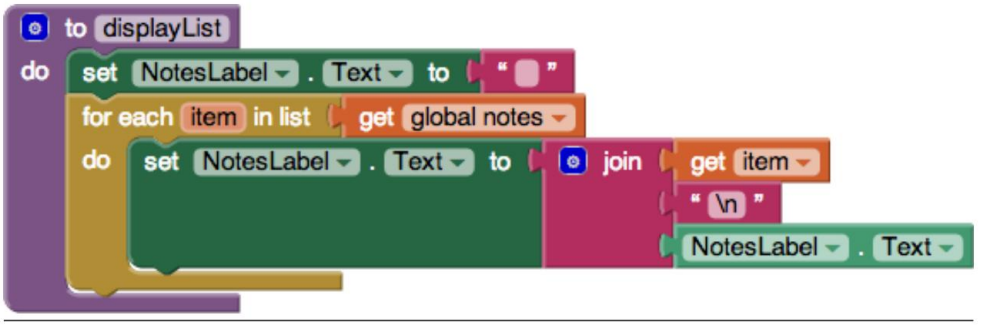
הגדרת נוהל

בואו נבנה נוהל לעשות את העבודה של בלוקי הקוד המיותרים מאיור 21-2ב-ppA, Inventor.
אתה מגדיר הליך באופן דומה לאופן שבו אתה מגדיר משתנים.
מהמגירה של נהלים, גרור החוצה או בלוק להליך או בלוק תוצאת להליך . השתמש באחרון אם
ההליך שלך אמור לחשב ערך כלשהו ולהחזיר אותו (נדון בגישה זו מעט מאוחר יותר בפרק).

לאחר גרירת גוש אל נוהל , תוכל לשנות את שם ברירת המחדל שלו על ידי לחיצה על המילה
"פרוצדורה" והקלדת שם חדש. הבלוקים המיותרים שברצונך לשחזר מבצעים את העבודה של
הצגת רשימה, אז נקרא לפרוצדורה displayList, המוצג באיור 21-3.

איור 21-4. לחץ על "הליך" כדי לתת שם להליך שלך

השלב הבא הוא הוספת הבלוקים בתוך ההליך. במקרה זה, אנו משתמשים בלוקים שכבר
קיימים, אז נגרור את אחד מהבלוקים המיותרים המקוריים ממטפל האירועים שלו ונמקם אותו
בתוך הבלוק to displayList, כפי שמוצג באיור 21-4.

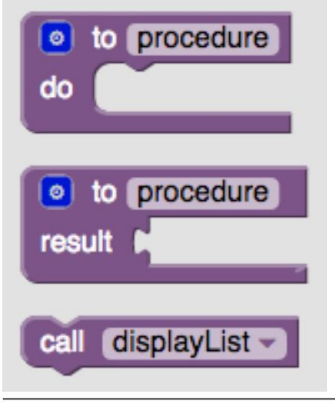


איור .21-5הליך displayListמקפל את הקוד המיותר

קריאה לנוהל

נהלים, כמו displayListו"צחצוח שיניים", הם ישויות עם פוטנציאל לבצע משימה. עם זאת, הם יבצעו את המשימה הזו רק אם הם יידרשו לעשות זאת. עד כה, יצרנו נוהל אך לא קראנו לו. לקרוא להליך פירושו להפעיל אותו, או לגרום לו לקרות.

ב-Inventor, ppAכאשר אתה מגדיר נוהל, הוספת חסימת שיחה אוטומטית למגירת הנהלים כפי שמוצג באיור .21-5



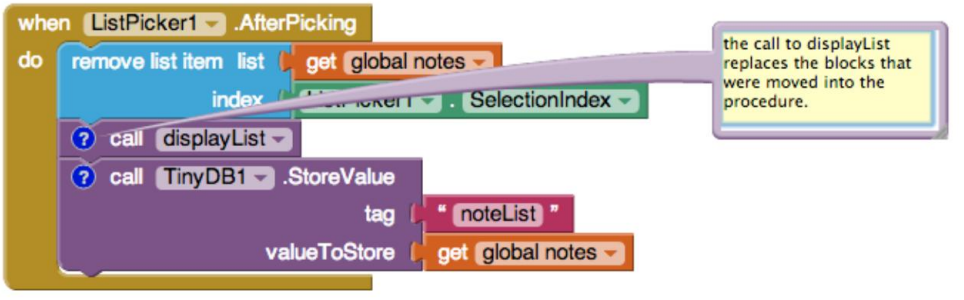
איור .21-6חסימת קריאה מופיעה במגירה של נהלים כאשר אתה מגדיר נוהל

כבר השתמשת בחסימות שיחות כדי להתקשר לפונקציות המוגדרות מראש של App Inventor, כגון Texting.SendMessage. -וBall.MoveToכאשר אתה מגדיר נוהל, בעצם יצרת בלוק משלך; הרחבת את שפת ממציא האפליקציות.

באמצעות חסימת השיחה החדשה , אתה יכול להפעיל את היצירה שלך.

לדוגמא של אפליקציית Note Taker, תגרור שלושה בלוקים של שיחות displayList ולהשתמש בהם כדי להחליף את הקוד המיותר בשלושת מטפלי האירועים. למשל, ה

יש לשנות את המטפל באירוע ListPicker1.AfterPicking(למחיקת הערה) כפי שמוצג באיור 21-6.



איור 21-7. שימוש בקריאה displayListכדי להפעיל את החסימות כעת בהליך

מונה התוכניות

כדי להבין כיצד פועל בלוק השיחה , חשבו על אפליקציה כבעלת מצביע שעובר דרך הבלוקים שמבצעים פונקציות. במדעי המחשב, מצביע זה נקרא מונה התוכניות.

כאשר מונה התוכנית מבצע את החסימות בתוך מטפל באירועים והוא מגיע לבלוק קריאה , הוא קופץ לפרוצדורה ומבצע את החסימות שבו. כאשר ההליך מסתיים, מונה התוכנית קופץ חזרה למיקומו הקודם (גוש השיחה) וממשיך משם. אז, עבור הדוגמה של Note Taker, בלוק הסר פריט רשימה מבוצע; לאחר מכן מונה התוכנית קופץ לנוהל displayListומבצע את הבלוקים בהליך זה (מגדיר את NotesLabel.Textלטקסט הריק, ואת ה- עבור כל אחד מהם); ולבסוף מונה התוכנית חוזר לבצע את בלוק TinyDB1.StoreValue .

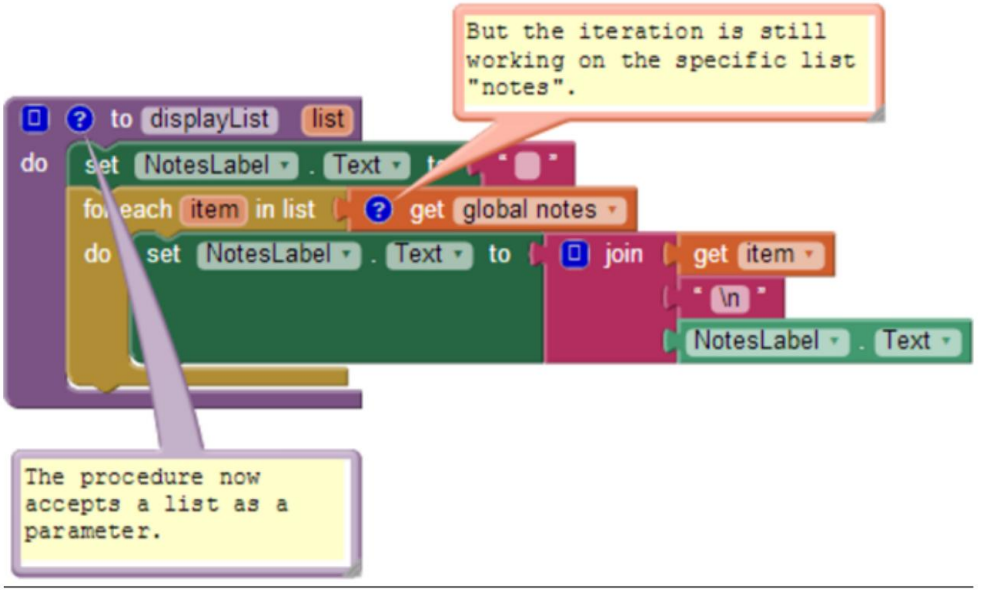
הוספת פרמטרים לנוהל שלך

הליך displayListמאפשר להחזיר קוד מיותר למקום אחד. קל יותר להבין את האפליקציה מכיוון שאתה יכול לקרוא את מטפלי האירועים ברמה גבוהה ובאופן כללי להתעלם מהפרטים של אופן הצגת רשימה. זה גם מועיל מכיוון שאתה עשוי להחליט לשנות את אופן הצגת הרשימה, וההליך מאפשר לך לבצע שינוי כזה במקום אחד (במקום שלושה).

עם זאת, לנוהל displayListיש מגבלות מבחינת התועלת הכללית שלו. ההליך פועל רק עבור רשימה ספציפית (הערות) ומציג רשימה זו בתווית ספציפית (NotesLabel). לא יכולת להשתמש בו כדי להציג רשימת נתונים שונה - לדוגמה, רשימה של משתמשי האפליקציה - מכיוון שהיא מוגדרת באופן ספציפי מדי.

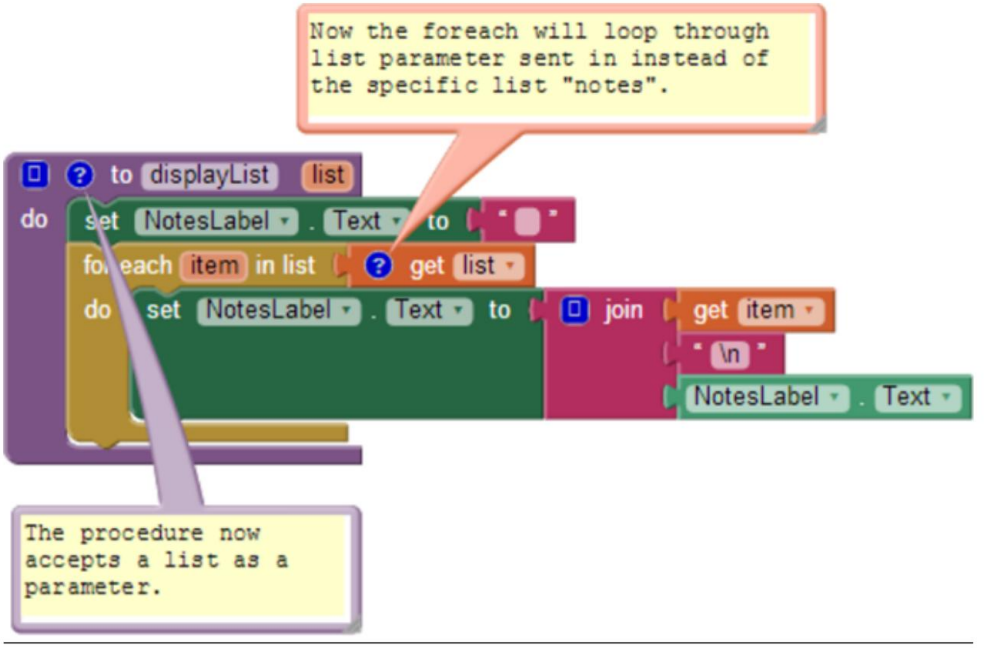
App Inventorושפות אחרות מספקות מנגנון שנקרא פרמטרים להפיכת נהלים לשימוש כלליים יותר. פרמטרים כוללים את המידע שהנוהל צריך כדי לבצע את עבודתו. הם מספקים את המפרטים של אופן ביצוע ההליך. בדוגמה שלנו לצחצוח שיניים לפני השינה, תוכל להגדיר "סוג משחת שיניים" ו"זמן צחצוח" כפרמטרים של הליך "צחצוח שיניים".

אתה מגדיר פרמטרים להליך על ידי לחיצה על הסמל הכחול בפינה השמאלית העליונה של הגדרת הנוהל. עבור הליך , displayListנגדיר פרמטר בשם "רשימה", כפי שמוצג באיור .21-7



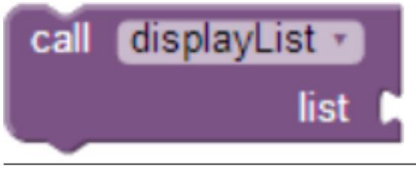
איור .21-8ההליך מקבל כעת רשימה כפרמטר

אפילו כשהפרמטר מוגדר, הבלוקים עדיין מתייחסים ישירות לרשימה הספציפית הערות (הוא מחובר לחריץ "ברשימה" של עבור כל אחד מהם). מכיוון שאנו רוצים שהפרוצדורה תשתמש ברשימה שאנו שולחים כפרמטר, אנו מחליפים את ההפניה להערות גלובליות בהפניה לקבלת רשימה, כפי שמודגם באיור .21-8



איור 21-9. כעת עבור כל אחד ישתמשו ברשימה שנשלחה

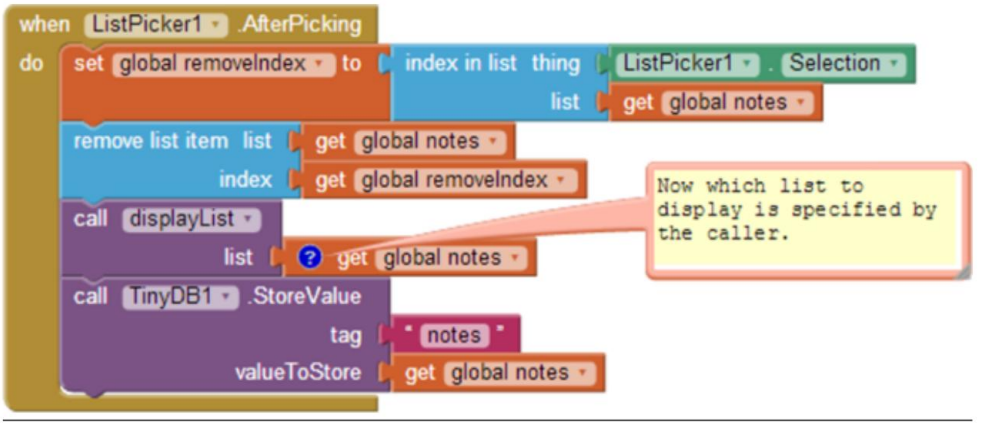
הגרסה החדשה של הנוהל גנרית יותר: קריאות ל- displayListיכולות כעת שלח לו רשימה כלשהי, ו- displayListיציג אותה. כאשר אתה מוסיף פרמטר לפרוצדורה, App Inventorמכניס אוטומטית שקע מתאים בבלוק השיחה .
לכן, כאשר רשימת הפרמטרים מתווספת ל- displayList, בלוקי הקריאה ל- displayListנראים כמו איור 21-9.



איור 21-10. קריאה displayList-למחייבת כעת לציין איזו רשימה להציג

רשימת הפרמטרים בתוך הגדרת הפרוצדורה נקראת פרמטר פורמלי.
השקע המתאים בתוך בלוק השיחה נקרא פרמטר ממשי. כאשר אתה קורא לנוהל ממקום כלשהו באפליקציה, עליך לספק פרמטר בפועל עבור כל פרמטר פורמלי של ההליך. אתה עושה זאת על ידי זרימת כל השקעים בשיחה.

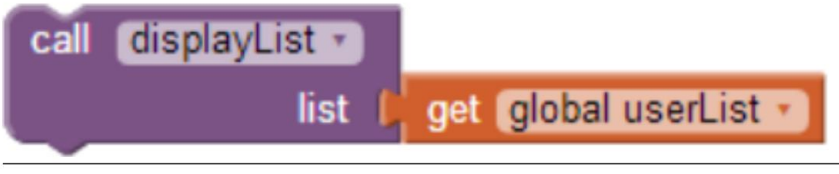
עבור אפליקציית Note Taker, אתה מוסיף הפניה לקבלת הערות גלובליות כפרמטר בפועל. איור 21-10 מראה כיצד יש לשנות את ListPicker.AfterSelection .



איור 21-11. קריאה displayList-לעם הערות שנשלחו כפרמטר בפועל

כעת, כאשר נקרא displayList, הערות הרשימה נשלחות אל הפרוצדורה וממוקמות ברשימת הפרמטרים. מונה התוכנית ממשיך לבצע את הבלוקים בפרוצדורה, תוך התייחסות לרשימת הפרמטרים אך באמת עובד עם המשתנה הערות.

בגלל הפרמטר, עכשיו אתה יכול להשתמש בהליך displayList עם כל רשימה, לא רק הערות. לדוגמה, אם אפליקציית Note Taker שותפה בין רשימת משתמשים וברצונך להציג את רשימת המשתמשים, תוכל להתקשר displayList-לולשלוח לה את userList, כפי שמוצג באיור 21-11.



איור 21-12. כעת ניתן להשתמש בהליך displayList כדי להציג כל רשימה, לא רק הערות

החזרת ערכים מהליך

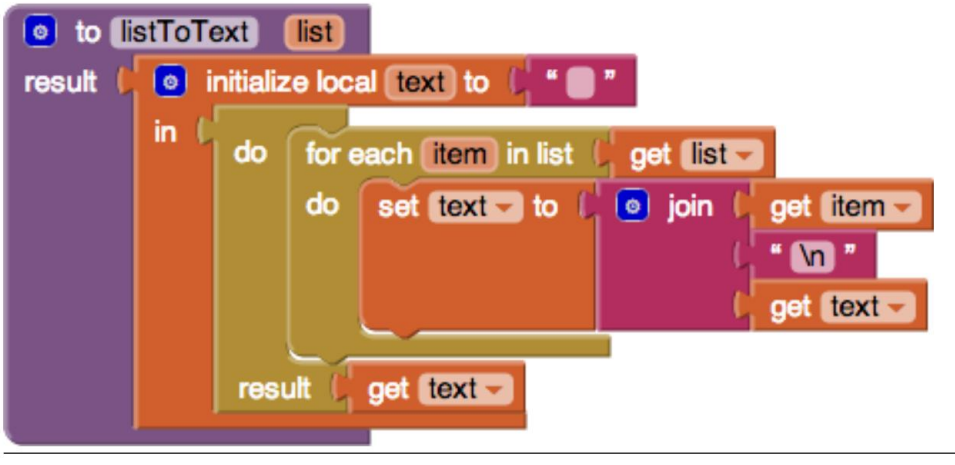
עדיין יש בעיה אחת בהליך displayList מבחינת התועלת הכללית שלו - האם אתה יכול להבין מה זה? כפי שזה כתוב כרגע, זה יכול להציג כל רשימה של נתונים, אבל זה תמיד יציג את הנתונים האלה בתווית NotesLabel. מה אם תרצה שהרשימה תוצג באובייקט ממשק משתמש אחר (למשל, הייתה לך תווית שונה להצגת ה- userList)?

פתרון אחד הוא להגות מחדש את ההליך ולשנות את תפקידו מהצגת רשימה בתווית מסוימת לפשוט החזרת אובייקט טקסט שאתה יכול

להציג בכל מקום. לשם כך, אתה משתמש בבלוק תוצאת פרוצדורה , המתואר באיור
21-12, במקום בלוק הפרוצדורה .

איור 21-13. חסימת תוצאת ההליך

אתה תבחין שבהשוואה לבלוק ההליך , תוצאת ההליך
לבלוק יש שקע נוסף בתחתית. אתה מציב משתנה במשבצת הזו והוא מוחזר למתקשר. לכן,
בדיוק כפי שהמתקשר יכול לשלוח נתונים לפרוצדורה עם פרמטר, פרוצדורה יכולה לשלוח
נתונים בחזרה עם ערך החזרה.
איור 21-13 מציג את הגרסה המחודשת של ההליך הקודם, הפעם באמצעות בלוק תוצאת
פרוצדורה . שימו לב שמכיוון שההליך עושה כעת עבודה שונה, השם שלו משתנה מ-
displayListל- listToText.



איור 21-14. listToText מחזיר אובייקט טקסט שהמתקשר יכול למקם בכל תווית

בבלוקים המוצגים באיור 21-13, טקסט משתנה מקומי מוגדר להחזיק את הנתונים
כאשר ההליך חוזר על כל פריט ברשימה. טקסט מאותחל כמשתנה מקומי, במקום משתנה
גלובלי, מכיוון שהוא משמש רק בהליך זה.
משתנה טקסט זה מחליף את רכיב NotesLabel הספציפי מדי שהיה
בשימוש בגרסת displayList של הליך זה. כאשר עבור כל אחד מסתיים, הטקסט המשתנה
מכיל את פריטי הרשימה, כאשר כל פריט מופרד על ידי תו שורה חדשה, \n (למשל,
"item1\nitem2\item3"). משתנה טקסט זה מחובר לאחר מכן לשקע ערך ההחזרה.

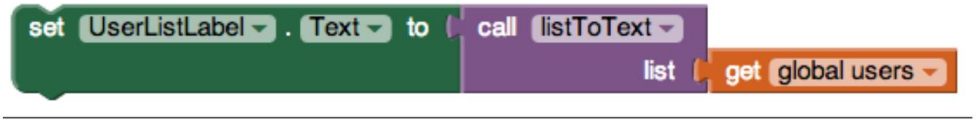
כאשר תוצאת הליך מוגדרת, בלוקי השיחה המתאימים לה נראים שונים
מאשר אלה עבור הליך. השווה את הקריאה ל- listToTextעם הקריאה ל- displayList
באיור .21-14

איור .21-15הקריאה מימין מחזירה ערך ולכן חייבת להיות מחוברת למשהו

ההבדל הוא שלרשימת השיחותtxeToT יש תקע בצד שמאל. הסיבה לכך היא שכאשר
השיחה מבוצעת, ההליך יעבור את המשימה שלו ואז יחזיר ערך לבלוק הקריאה . ערך ההחזר
הזה חייב להיות מחובר למשהו.
במקרה זה, המתקשרים ל- displayListיכולים לחבר את ערך ההחזר הזה לכל תווית שהם
רוצים. עבור הדוגמה של ,Note Takerשלושת מטפלי האירועים שצריכים להציג רשימה
יתקשרו לפרוצדורה, כפי שמוצג באיור .21-15

איור .21-16המרת הערות הרשימה לטקסט והצגתן NotesLabel-ב

הנקודה החשובה כאן היא שמכיוון שההליך הוא גנרי לחלוטין ו
אינו מתייחס לרשימות או תוויות כלשהן באופן ספציפי, חלק אחר של האפליקציה יכול
להשתמש בו כדי להציג כל רשימה בתווית כלשהי, כפי שמוצג באיור .21-16



איור .21-17ההליך אינו קשור עוד לרכיב תווית מסוים

שימוש חוזר בלוקים בין אפליקציות

אין צורך להגביל שימוש חוזר בלוקי קוד באמצעות נהלים לאפליקציה אחת. ישנם נהלים רבים,
כגון ,listToTextשבהם תוכל להשתמש כמעט בכל אפליקציה שתיצור. בפועל, ארגונים וקהילות
תכנות בונים ספריות קוד של נהלים עבור תחומי העניין שלהם.

בדרך כלל, שפות תכנות מספקות כלי ייבוא שדרכו ניתן לכלול קוד ספרייה בכל אפליקציה. ל-ppA Inventorעדיין אין כלי עזר כזה. הדרך היחידה לשתף נהלים היא ליצור אפליקציית ספרייה מיוחדת ולהתחיל בפיתוח אפליקציה חדשה על ידי שמירת עותק חדש של אותה אפליקציה ועבודה ממנה.

נוהל המרחק בין נקודות

עם הדוגמה של , displayList (listToText)אפיינו את הגדרת הפרוצדורה כדרך לחסל קוד מיותר: אתה מתחיל לכתוב קוד, מוצא יתירות תוך כדי, ומשנה את הקוד שלך כדי לחסל אותם. בדרך כלל, עם זאת, מפתח תוכנה או צוות יעצבו אפליקציה מההתחלה תוך מחשבה על נהלים וחלקים לשימוש חוזר. תכנון מסוג זה יכול לחסוך לך זמן משמעותי ככל שהפרויקט מתקדם.

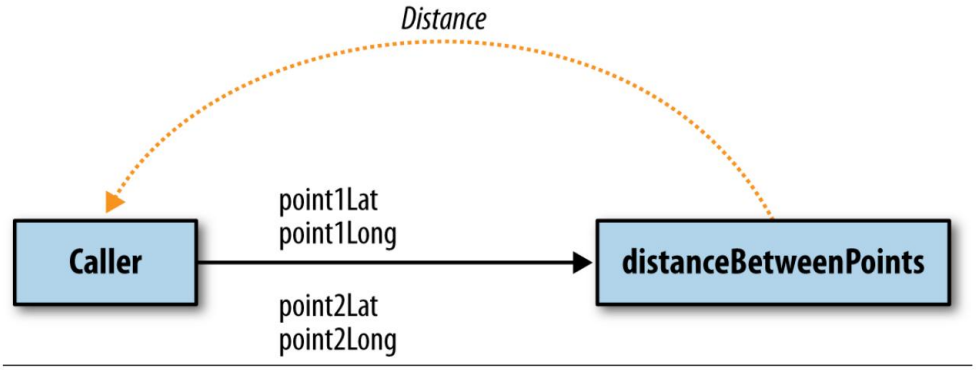
שקול אפליקציה כדי לקבוע את בית החולים המקומי הקרוב ביותר למיקומו הנוכחי של המשתמש -משהו שיהיה שימושי מאוד במקרה חירום. להלן תיאור עיצוב ברמה גבוהה של האפליקציה:

כאשר האפליקציה מופעלת, מצא את המרחק, במיילים, בין המיקום הנוכחי לבית החולים הראשון. ואז מצא אותו לבית החולים השני, וכן הלאה. כאשר יש לך את המרחקים, קבע את המרחק המינימלי והצג את הכתובת (ו/או מפה) למיקום זה.

מהתיאור הזה, האם אתה יכול לקבוע את ההליכים שהאפליקציה הזו צריכה? לעתים קרובות, הפעלים בתיאור כזה מרמזים על ההליכים שתזדקקו להם. חזרה בתיאור שלך, כפי שצוין עם ה-"וכן הלאה," הוא רמז נוסף. במקרה זה, איתור המרחק בין שתי נקודות וקביעת המינימום של מרחקים מסוימים הם שני הליכים הכרחיים.

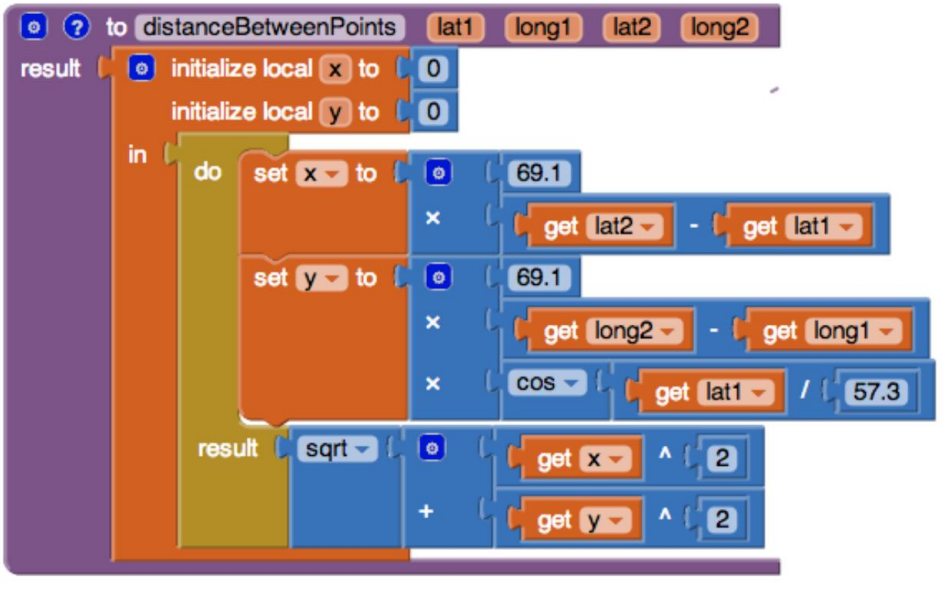
בוא נחשוב על העיצוב של הפרוצדורה לאיתור המרחק בין שתי נקודות, שאותן נקרא ' מרחק בין נקודות' (enf, אז מקוריות היא לא הצד החזק שלי). בעת תכנון פרוצדורה, צריך לקבוע את התשומות והפלטים שלו: הפרמטרים שהמתקשר ישלח להליך כדי שיעשה את עבודתו ואת ערך התוצאה שההליך ישלח בחזרה למתקשר. במקרה זה, המתקשר צריך לשלוח את קו הרוחב והאורך של שתי הנקודות להליך, כפי שמוצג באיור .21-17

תפקידו של הנוהל הוא להחזיר את המרחק, בקילומטרים.



איור 21-18. המתקשר שולח ארבעה פרמטרים קלט ומקבל מרחק

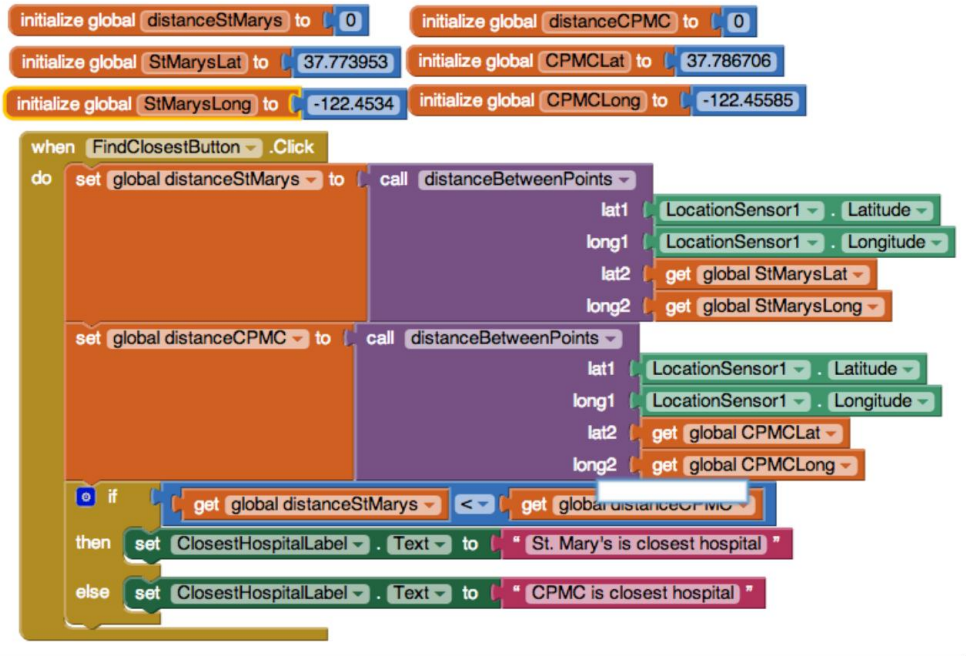
איור 21-18 מציג את ההליך שנתקלנו בו בתחילת הפרק,
שימוש בנוסחה לקירוב הקילומטראז' בין שתי קואורדינטות GPS.



איור 21-19. נוהל distanceBetweenPoints

איור 21-19 מציג בלוקים שמבצעים שתי קריאות לנוהל, כל אחד מהם
מאתר את המרחק מהמיקום הנוכחי לבית חולים מסוים.
עבור השיחה הראשונה, הפרמטרים בפועל עבור הנקודה הראשונה הם הקריאות
הנוכחיות מ- LocationSensor, בעוד שהנקודה השנייה מוגדרת על ידי קואורדינטות
ה-SPG עבור בית החולים St. Mary's. הערך המתקבל ממוקם במשתנה
distanceStMarys. השיחה השנייה דומה אך במקום זאת משתמשת בנתונים עבור בית
החולים CPMC עבור הנקודה השנייה.

האפליקציה ממשיכה להשוות את שני המרחקים שהוחזרו כדי לקבוע איזה בית חולים הכי קרוב. אבל, אם היו יותר בתי חולים מעורבים, באמת היית צריך להשוות רשימה של מרחקים כדי למצוא את הקצר ביותר. ממה שלמדת, האם אתה יכול ליצור הליך בשם findMinimum שמקבל רשימת מספרים כפרמטר ומחזיר את האינדקס של המינימום?



איור 21-20. שתי קריאות לנוהל distanceBetweenPoints

סיכום

שפות תכנות כגון App Inventor מספקות סט בסיסי של פונקציונליות מובנית. באמצעות שימוש בפרוצדורות, ממציאי אפליקציות יכולים להרחיב את השפה הזו עם הפשטות חדשות. App Inventor לא מספק חסימה להצגת רשימה, אז אתה בונה אחת. צריך בלוק לחישוב המרחק בין קואורדינטות GPS?
אתה יכול ליצור משלך.

היכולת להגדיר בלוקים של פרוצדורות ברמה גבוהה יותר היא המפתח להנדסה גדולה, תוכנה הניתנת לתחזוקה ופתרון בעיות מורכבות מבלי להיות מוצף כל הזמן מכל הפרטים. נהלים מאפשרים לך לכלול בלוקים של קוד ולתת שם לבלוקים האלה. בזמן שאתה מתכנת את ההליך, אתה מתמקד אך ורק בפרטים של אותם בלוקים. עם זאת, בתכנות שאר האפליקציה, כעת יש לך הפשטה -שם -שאתה יכול להתייחס אליו ברמה גבוהה.

עבודה עם מאגרי מידע

איור 22-1.

לפייסבוק יש מאגר מידע של כל חבר פרטי חשבון, רשימת חברים ופוסטים. לאמזון יש מסד נתונים של כמעט כל מה שאתה יכול לקנות. לגוגל יש מסד נתונים של מידע על כל עמוד ברשת העולמית. למרות שלא בקנה מידה כזה, כמעט כל אפליקציה לא טריוויאלית שתוכל ליצור תיצור אינטראקציה עם מסד נתונים.

ברוב סביבות התכנות, בניית אפליקציה שמתקשרת עם מסד נתונים היא טכניקת תכנות מתקדמת: צריך להגדיר שרת עם תוכנת מסד נתונים כמו Oracle או MySQL ואז לכתוב קוד שמתממשק לאותו מסד נתונים. באוניברסיטאות רבות, תכנות מסד נתונים כזה לא נלמד עד לקורס הנדסת תוכנה או מסדי נתונים ברמה עליונה.

כשזה מגיע למאגרי מידע, App Inventor עושה עבורך את המשימות הכבדות (והרבה דברים שימושיים אחרים!). השפה מספקת רכיבים שמצמצמים תקשורת מסד נתונים לפעולות אחסון וקבלת פשוטות. אתה יכול ליצור אפליקציות המאחסנות נתונים ישירות במכשיר האנדרואיד, ועם הגדרות מסוימות, אתה יכול ליצור אפליקציות המשתפות נתונים עם מכשירים ואנשים אחרים על ידי אחסונם במסד נתונים מרכזי ברשת.
הנתונים המאוחסנים במשתנים ובמאפייני הרכיבים הם קצרי טווח: אם המשתמש מקליד מידע מסוים בטופס ולאחר מכן סוגר את האפליקציה לפני שהמידע הזה נשמר במסד נתונים, המידע ייעלם כאשר האפליקציה תיפתח מחדש. כדי לאחסן מידע באופן קבוע, עליך לאחסן אותו במסד נתונים. אומרים שהמידע במאגרי מידע הוא מתמיד מכיוון שגם כאשר סוגרים את האפליקציה ופותחים אותה מחדש, הנתונים עדיין זמינים.

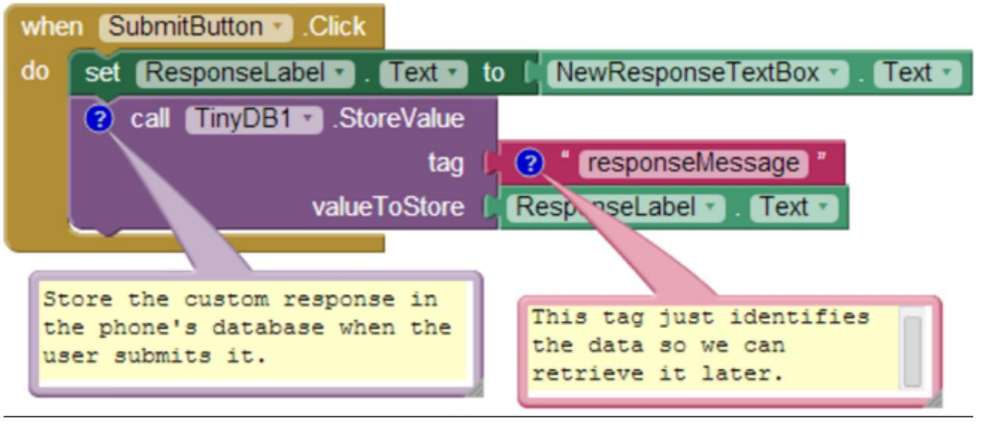
כדוגמה, קחו בחשבון את אפליקציית No Texting While Driving של פרק 4, אשר שולחת תגובה אוטומטית להודעות טקסט נכנסות. לאפליקציה יש תגובת ברירת מחדל שנשלחת, אך היא מאפשרת למשתמש להזין הודעה מותאמת אישית שתישלח, במקום זאת. אם המשתמש משנה את ההודעה המותאמת אישית ל"אני ישן; תפסיק להציק לי" ואז סוגר את האפליקציה, ההודעה עדיין צריכה להיות "אני ישן; תפסיק להציק לי", ולא ברירת המחדל המקורית, כאשר האפליקציה נפתחת מחדש. לפיכך, ההודעה המותאמת אישית חייבת להיות מאוחסנת במסד נתונים, ובכל פעם שהאפליקציה נפתחת, יש לאחזר את ההודעה הזו מהמסד חזרה לאפליקציה.

אחסון נתונים מתמשכים ב-TinyDB

App Inventor מספק שני רכיבים כדי להקל על פעילות מסד הנתונים: TinyDB ו- TinyWebDB. אתה משתמש ב- TinyDB כדי לאחסן נתונים קבועים ישירות במכשיר האנדרואיד; זה שימושי עבור אפליקציות אישיות שעבורן המשתמש לא יצטרך לשתף נתונים עם מכשיר או אדם אחר, כמו ב'אין לשלוח הודעות טקסט בזמן נהיגה'. מצד שני, אתה משתמש ב- TinyWebDB כדי לאחסן נתונים במסד נתונים אינטרנטי שניתן לשתף בין מכשירים. היכולת לגשת לנתונים ממסד נתונים אינטרנטי חיונית למשחקים ואפליקציות מרובי משתמשים שאיתם משתמשים יכולים להזין ולשתף מידע (כמו אפליקציית "MakeQuiz" בפרק 10).

רכיבי מסד הנתונים דומים, אבל TinyDB היא קצת יותר פשוטה, אז אנחנו נחקור את זה ראשון. עם TinyDB, אתה לא צריך להגדיר את מסד הנתונים בכלל; הנתונים מאוחסנים במסד נתונים ישירות במכשיר ומשויכים לאפליקציה שלך.

אתה מעביר נתונים לזיכרון לטווח ארוך עם בלוק TinyDB.StoreValue , as מוצג באיור 22-1, שמגיע מהאפליקציה ללא הודעות טקסט בזמן נהיגה.



איור 22-2. בלוק TinyDB.StoreValue מאחסן נתונים בזיכרון לטווח ארוך של המכשיר

סכימת תג-ערך משמשת לאחסון מסד נתונים. באיור 22-1, הנתונים מתויגים בטקסט "responseMessage". הערך הוא טקסט כלשהו שהמשתמש הקליד בתיבת טקסט עבור התגובה המותאמת אישית החדשה - משהו כמו, "אני ישן; תפסיק להטריד אותי."

פרמטר התג נותן לנתונים שאתה מאחסן במסד הנתונים שם, דרך להתייחס למידע. הערך הוא הנתונים עצמם. אתה יכול לחשוב על התג כמפתח שבו תשתמש מאוחר יותר כשתרצה לאחזר את הנתונים ממסד הנתונים.

באופן דומה , אתה יכול לחשוב על מסד נתונים של App Inventor TinyDB כטבלה של ערך תג זוגות. לאחר ביצוע ה- TinyDB1.StoreValue באיור 22-1, למסד הנתונים של ההתקן יהיה הערך המופיע בטבלה 22-1.

טבלה 22-1. הערך המאוחסן במסדי הנתונים

תג	ערך	
responseMessage	אני ישן; תפסיק להטריד אותי	

אפליקציה עשויה לאחסן צמדי תג-ערך רבים עבור פריטי הנתונים השונים שברצונך שיהיו מתמידים. התג הוא תמיד טקסט, בעוד שהערך יכול להיות פיסת מידע בודדת (טקסט או מספר) או רשימה. לכל תג יש רק ערך אחד; בכל פעם שאתה מאחסן לתג, הוא מחליף את הערך הקיים.

אחזור נתונים מ-BDyniT

אתה מחזיר נתונים ממסד הנתונים באמצעות בלוק TinyDB.GetValue. כאשר אתה מתקשר ל-GetValue, אתה מבקש נתונים מסוימים על ידי מתן תג. עבור האפליקציה ללא הודעות טקסט בזמן נהיגה, אתה יכול לבקש את התגובה המותאמת אישית באמצעות אותו תג שבו השתמשת ב-StoreValue, "responseMessage". הקריאה ל-GetValue מחזירה את הנתונים, לכן עליך לחבר אותם למשתנה.

לעתים קרובות, תוכל לאחזר נתונים ממסד הנתונים כאשר האפליקציה נפתחת. Inventor מספק מטפל מיוחד באירועים, Screen.Initialize, המופעל כאשר האפליקציה מופעלת. עליך להקפיד לשקול את המקרה כאשר עדיין אין נתונים במסד הנתונים (למשל, האפליקציה הראשונה שבה מופעלת). כאשר אתה קורא ל-GetValue, אתה מציין פרמטר valueIfTagNotThere. אם אין נתונים, הערך הזה יוחזר מהשיחה.

הבלוקים באיור 22-2, עבור המסך. אתחול של אי שליחת הודעות טקסט בזמן נהיגה מעידים על הדרך שבה אפליקציות רבות טוענות נתוני מסד נתונים בעת האתחול. app, הבלוקים מכניסים את הנתונים שהוחזרו מ- GetValue לתוויות ResponseLabel. אם יש נתונים כבר במסד הנתונים, הם ממוקמים ב- ResponseLabel. אם אין נתונים עבור התג הנתון, הערך valueIfTagNotThere "אני נוהג עכשיו, אשלח לך טקסט מאוחר יותר" במקרה זה, ממוקם ב- ResponseLabel.

איור 22-3. כאשר האפליקציה מופעלת, לעתים קרובות תשלוף מידע על מסד הנתונים

נתונים משותפים ו-BDbeWyniT

רכיב TinyDBמאחסן נתונים במסד נתונים הממוקם ישירות במכשיר האנדרואיד. זה מתאים לאפליקציות לשימוש אישי שאינן צריכות לשתף נתונים בין משתמשים. לדוגמה, אנשים רבים עשויים להתקין את האפליקציה ללא הודעות טקסט בזמן נהיגה, אך אין צורך שהאנשים השונים המשתמשים באפליקציה ישתפו את התגובות המותאמות אישית שלהם עם אחרים.

כמובן, אפליקציות רבות אכן חולקות נתונים: חשבו על פייסבוק, טוויטר ומשחקים מרובי משתמשים. עבור אפליקציות לשיתוף נתונים כאלה, מסד הנתונים חייב להיות ברשת, לא במכשיר, כך שמשתמשי אפליקציה שונים יוכלו לתקשר איתו ולגשת למידע שלו.

TinyWebDBהיא המקבילה האינטרנטית ל- .TinyDBעם זה, אתה יכול לכתוב אפליקציות שחנות נתונים באינטרנט, באמצעות פרוטוקול StoreValue/GetValueדומה לזה של .TinyDB כברירת מחדל, רכיב TinyWebDBמאחסן נתונים באמצעות מסד נתונים אינטרנט שהוגדר על ידי צוות App Inventorונגיש בכתובת .http://appinvtinywebdb.appspot.comאתר זה מכיל מסד נתונים ומגיש (מגיב) בקשות אינטרנט לאחסון ואחזור נתונים. האתר מספק גם ממשק אינטרנט קריא אנושי שמנהל מסד נתונים (אתה) יכול להשתמש בו כדי לבחון את הנתונים המאוחסנים בו.

מסד נתונים ברירת מחדל זה מיועד לפיתוח בלבד; הוא מוגבל בגודלו ונגיש ל כל מתכנתי .App Inventorמכיוון שכל אפליקציית App Inventorיכולה לאחסן שם נתונים, אין לך ביטחון שאפליקציה אחרת לא תחליף את הנתונים שלך!

אם אתה רק בוחן את App Inventorאו בשלבים מוקדמים של פרויקט, מסד הנתונים האינטרנטי המוגדר כברירת מחדל הוא .fneאבל, אם אתה יוצר אפליקציה לפריסה בעולם האמיתי, בשלב מסוים תצטרך להגדיר מסד נתונים אינטרנט משלך. מכיוון שאנחנו רק חוקרים עכשיו, נשתמש במסד הנתונים המוגדר כברירת מחדל. בהמשך הפרק, תלמד כיצד ליצור מסד נתונים אינטרנט משלך ולהגדיר את TinyWebDBלהשתמש בו במקום זאת.

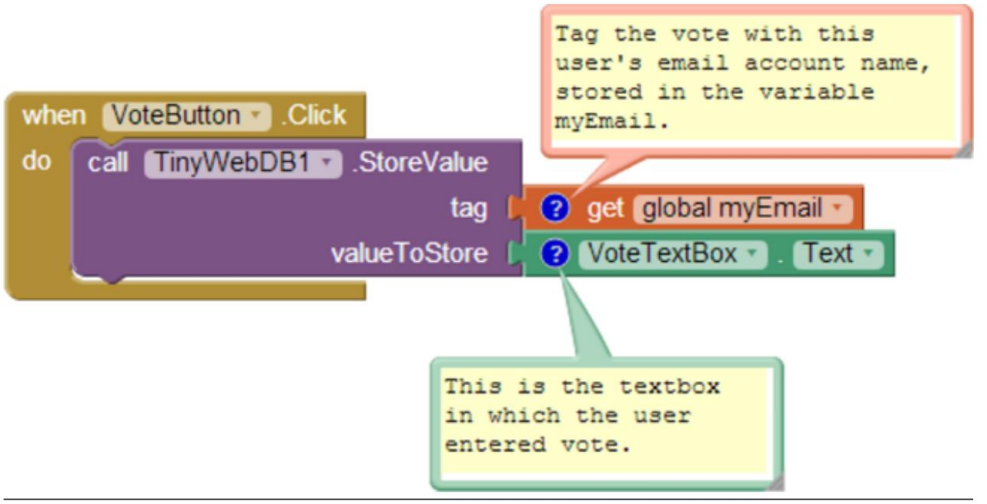
בחלק זה, נבנה אפליקציית הצבעה (מתוארת באיור 22-3כדי להמחיש כיצד TinyWebDBעובד. לאפליקציה יהיו התכונות הבאות:

- משתמשים מתבקשים להזין את כתובת האימייל שלהם בכל פעם שהאפליקציה נטענת. זֶה שם החשבון ישמש לתיוג ההצבעה של המשתמש במסד הנתונים.
- משתמשים יכולים להגיש הצבעה חדשה בכל עת. במקרה זה, ההצבעה הישנה שלהם תהיה מוחלף.
- משתמשים יכולים לראות את ההצבעות של כל אחד בקבוצה.
- לשם הפשטות, הנושא עליו מצביעים נקבע מחוץ לאפליקציה, כמו למשל במסגרת כיתתית בה המורה מכריזה על הנושא ומבקשת מכולם להצביע באופן אלקטרוני. (שים לב שניתן להרחיב את הדוגמה הזו כדי לאפשר למשתמשים להצביע על ידי פרסום נושאים להצבעה מתוך האפליקציה.)

איור 22-4. אפליקציית הצבעה המאחסנת הצבעות ל-BDbeWyniT

אחסון נתונים על ידי שימוש ב-BBDEWYNIT

בלוק TinyWebDB.StoreValue פועל באותו אופן כמו TinyDB.StoreValue, מלבד העובדה שהנתונים מאוחסנים באינטרנט. עבור מדגם ההצבעה שלנו, נניח שהמשתמש יכול להזין הצבעה בתיבת טקסט בשם VoteTextBox ולהקיש על כפתור בשם VoteButton כדי לשלוח את ההצבעה. לאחסן את ההצבעה במאגר האינטרנט כך שאחרים יכול לראות את זה, נקודד את VoteButton.Click מטפל באירועים כמו הדוגמה באיור 22-4.



איור 22-5. כאשר המשתמש מזין הצבעה, היא מאוחסנת במסד הנתונים האינטרנטי

התג המשמש לזיהוי הנתונים הוא האימייל של המשתמש, אשר נשמר בעבר במשתנה myEmail (אתה תראה זאת מאוחר יותר). הערך הוא מה שהמשתמש הקליד ב-xoBtxeTetoV. לכן, אם האימייל של המשתמש היה joe@zmail.com וההצבעה שלו הייתה "פיצה", הערך יישמר במסד הנתונים כפי שמוצג בטבלה 22-2.

טבלה 22-2. התג והערך להצבעה נרשמים במסד הנתונים

תָּג	עֵרֶך
joe@zmail.com	פיצה

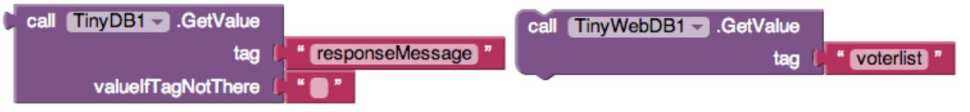
בלוק TinyWebDB.StoreValue שולח את צמד הערך-תג דרך האינטרנט לשרת מסד הנתונים בכתובת http://appinvtinywebdb.appspot.com. בזמן שאתה בודק את האפליקציה שלך, אתה יכול לעבור לכתובת האתר הזו, ללחוץ על getValue ולהזין תג שעבורו אחסנת ערך. האתר יציג לך את הערך הנוכחי של התג הזה.

בקשה ועיבוד נתונים עם TINYWEDBB

אחזור נתונים עם TinyWebDB מסובך יותר מאשר עם TinyDB. עם TinyDB, פעולת GetValue מחזירה מיד ערך מכיוון שהאפליקציה שלך מתקשרת עם מסד נתונים ישירות במכשיר האנדרואיד. עם TinyWebDB, האפליקציה מבקשת נתונים דרך האינטרנט, מה שעשוי לקחת זמן, כך ש-diordnA דורשת סכימה דו-שלבית לטיפול בה.

עם TinyWebDB, קריאה ל- GetValue מבקשת רק את הנתונים; זה באמת צריך להיקרא "RequestValue" מכיוון שהוא פשוט מגיש את הבקשה למסד הנתונים האינטרנטי ולא עושה זאת

למעשה לקבל ממנו ערך מיד. כדי לראות זאת בצורה ברורה יותר, בדוק את ההבדל בין בלוק TinyDB.GetValue לבלוק TinyWebDB.GetValueהמוצג באיור 22-5.



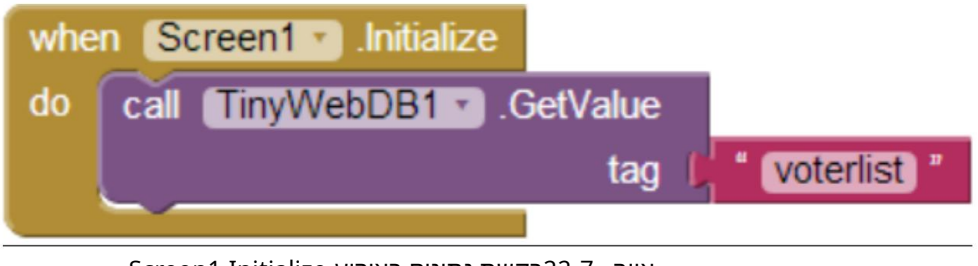
איור 22-6. החסימות TinyDB.GetValue ו-eulaVtoG.BDyniT

בלוק TinyDB.GetValueמחזיר ערך מיד, וכך מופיע פלאג בצדו השמאלי כך שניתן להכניס את הערך המוחזר למשתנה או תכונה. הבלוק TinyWebDB.GetValueאינו מחזיר ערך באופן מיידי, ולכן אין תקע בצד שמאל שלו.

במקום זאת, כאשר מסד הנתונים האינטרנטי ממלא את הבקשה והנתונים חוזרים ל-התקן, מופעל אירוע . TinyWebDB.GotValueאז, תתקשר TinyWebDB.GetValue-לבמקום אחד באפליקציה שלך, ולאחר מכן תתכנת את מטפל האירועים TinyWebDB.GotValueכדי לציין כיצד לטפל בנתונים כשהם מגיעים בפועל. מטפל באירועים כגון TinyWebDB.GotValue נקרא לפעמים הליך התקשרות חוזרת, מכיוון שישות חיצונית כלשהי (מסד הנתונים האינטרנטי) פועלת ומתקשרת לאפליקציה שלך בחזרה לאחר עיבוד הבקשה שלך. זה דומה להזמנה בבית קפה עמוס: אתה מבצע את ההזמנה שלך ואז מחכה שהבריסטה יקרא בשמך כדי ללכת לאסוף את המשקה שלך. בינתיים היא גם קיבלה הזמנות מכל השאר בתור (וגם האנשים האלה מחכים שיקראו בשמותיהם).

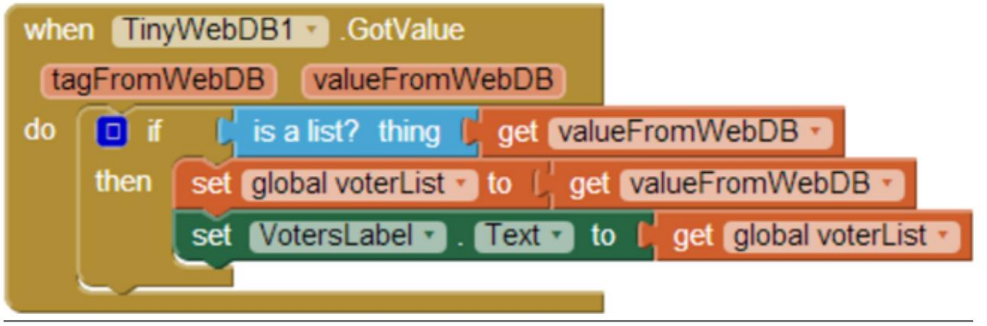
GETVALUE-GOTVALUEבפעולה

עבור האפליקציה לדוגמה שלנו, עלינו לאחסן ולאחזר רשימה של הבוחרים שיש להם את האפליקציה, מכיוון שהאפליקציה צריכה להציג את ההצבעות של כל המשתמשים.
הסכימה הפשוטה ביותר לאחזור נתוני רשימה היא לבקש את הנתונים כאשר האפליקציה מופעלת, באירוע , Screen.Initializeכפי שמוצג באיור 22-6.(בדוגמה זו, פשוט נקרא למסד הנתונים עם התג של "רשימת בוחרים".)



איור 22-7. בקשת נתונים באירוע Screen1.Initialize

כאשר רשימת הבוחרים מגיעה ממסד הנתונים האינטרנטי, המטפל באירועים TinyWebDB1.GotValueמופעל. איור 22-7מציג כמה בלוקים לעיבוד הרשימה המוחזרת.



איור 22-8. שימוש במטפל האירועים GotValueכדי לעבד את הרשימה שהוחזרה

הארגומנט valueFromWebDBשל GotValueמחזיק את הנתונים המוחזרים מבקשת מסד הנתונים. לארגומנטים של אירועים כגון valueFromWebDBיש משמעות רק בתוך מטפל האירועים שמפעיל אותם. הם נחשבים מקומיים למטפל באירועים, מכיוון שאינך יכול להתייחס אליהם במטפלי אירועים אחרים.

מכיוון שארגומנטים כגון valueFromWebDBאינם נגישים באופן גלובלי, אם אתה צריך את המידע בכל האפליקציה שלך, אתה צריך להעביר אותו למשתנה גלובלי. בדוגמה, התפקיד העיקרי של GotValueהוא להעביר את הנתונים המוחזרים ב- valueFromWebDBלתוך המשתנה ,voterListשבו תשתמש במטפל אחר באירועים.

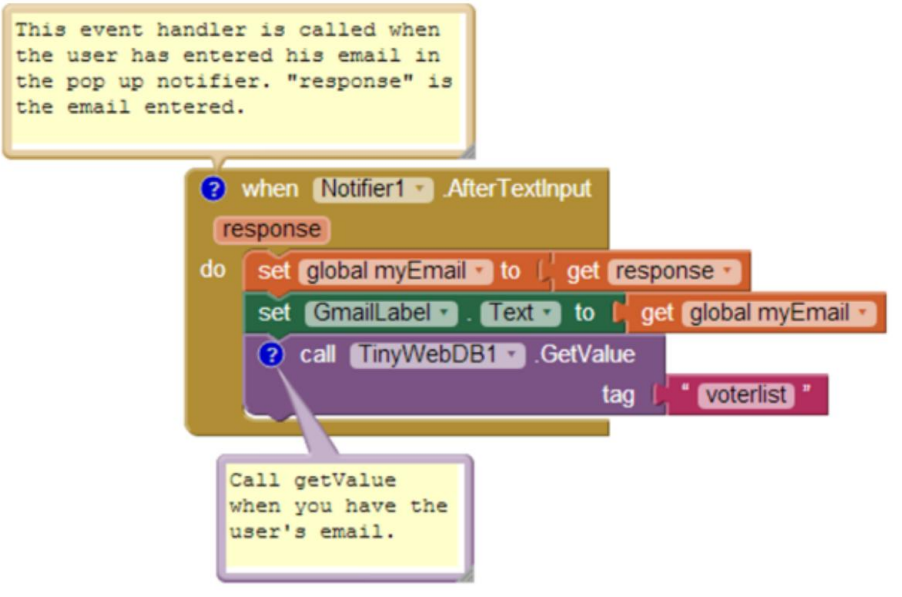
בלוק ifבמטפל באירועים משמש לעתים קרובות גם יחד עם GotValue, הסיבה היא שמסד הנתונים מחזיר טקסט ריק ("")ב- valueFromWebDBאם אין נתונים עבור התג המבוקש. ערך החזרה ריק זה מתרחש לרוב כאשר זו הפעם הראשונה שבה נעשה שימוש באפליקציה. אם תשאל אם ה- valueFromWebDBהוא רשימה, אתה מוודא שיש נתונים שהוחזרו בפועל. אם ה- valueFromWebDBהוא הטקסט הריק (מבחן ה- ifהוא שקר), אתה לא מכניס אותו ל- voterList.

דוגמה מורכבת יותר של GETVALUE/GOTVALUE

הבלוקים באיור 22-7הם מודל טוב לאחזור נתונים באפליקציה פשטנית למדי. אולם בדוגמה שלנו להצבעה, אנו זקוקים להיגיון מסובך יותר. באופן ספציפי:

- האפליקציה צריכה לבקש מהמשתמש להקליד כתובת דואר אלקטרוני עם תחילת התוכנית. אנחנו יכולים להשתמש ברכיב Notifierבשביל זה, שמקפיץ חלון. (תוכל למצוא את ה- Notifierבלוח "ממשק משתמש" ב-rengiseD.) כאשר המשתמש מקליד אימייל, נאחסן אותו במשתנה.
- רק לאחר קביעת האימייל של המשתמש יש להתקשר ל- GetValueכדי לאחזר את רשימת הבוחרים. אתה יכול להבין למה?

איור 22-8מציג את הבלוקים עבור תכנית מסובכת יותר זו לבקשת נתוני מסד הנתונים.



איור 22-9.בסכימה מורכבת יותר זו, GetValueנקרא לאחר קבלת האימייל של המשתמש במקום Screen.Initialize-ב

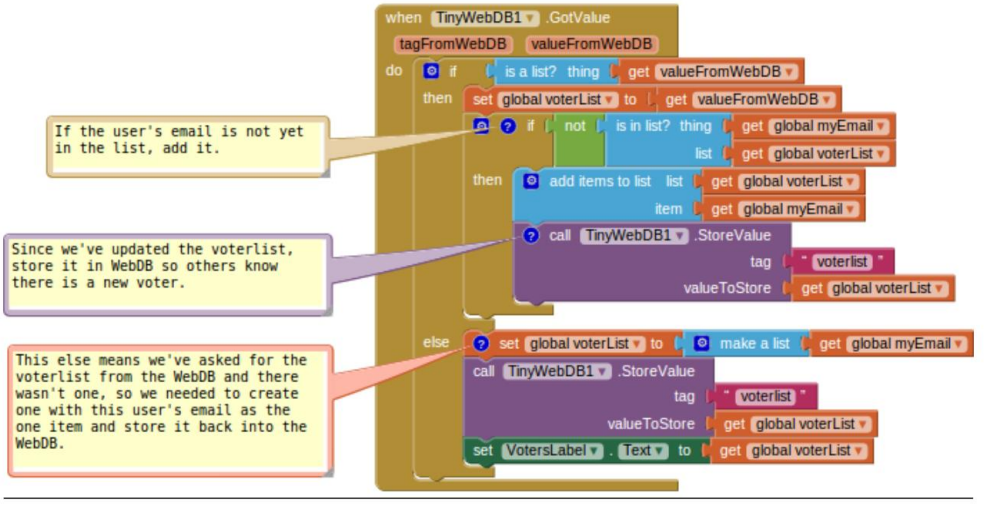
עם ההפעלה ,(Screen1.Initialize)רכיב Notifierמבקש מהמשתמש לבצע הקלד כתובת אימייל. כאשר המשתמש עושה זאת, והמטפל באירוע Notifier.AfterTextInput מופעל, הערך מוכנס למשתנה ולתווית, ואז נקרא GetValueכדי לקבל את רשימת הבוחרים. שימו לב ש- GetValueלא נקרא ישירות ב- ,Screen.Initializeמכיוון שאנו צריכים שכתובת הדוא"ל של המשתמש תוגדר תחילה.
לכן, עם החסימות האלה, כשהאפליקציה מאתחלת, היא מבקשת מהמשתמש להקליד כתובת דוא"ל ואז קורא ל- GetValueעם תג "רשימת בוחרים". כאשר הרשימה מגיעה מהאינטרנט, GotValueמופעלת. הנה מה שצריך לקרות:

GotValue •צריכה לבדוק אם הנתונים שמגיעים אינם ריקים (מישהו השתמש באפליקציה ויזם את רשימת הבוחרים). אם יש נתונים (רשימת בוחרים), GotValueצריכה לבדוק אם כתובת האימייל של המשתמש המסוים שלנו כבר נמצאת ברשימת הבוחרים. אם לא, יש להוסיף אותו לרשימה, ואת הרשימה המעודכנת יש לאחסן בחזרה למסד הנתונים.

•אם עדיין אין רשימת בוחרים במאגר, עלינו ליצור אחת עם רשימת הבוחרים של המשתמש כתובת דואר אלקטרוני כפריט היחיד.

איור 22-9מציג את הבלוקים להתנהגות זו.

הבלוקים שואלים תחילה אם רשימת בוחרים לא ריקה חזרה מהמאגר באמצעות קריאה היא רשימה? אם כן, הנתונים מוכנסים לרשימת הבוחרים המשתנה . זכור, לרשימת הבוחרים יהיו כתובות דוא"ל לכל מי שהשתמש באפליקציה הזו. עם זאת, אנחנו לא יודעים אם המשתמש הספציפי הזה עדיין ברשימה, אז אנחנו צריכים לבדוק. אם המשתמש עדיין לא ברשימה, כתובת הדוא"ל של המשתמש מתווספת עם הוסף פריט לרשימה, והרשימה המעודכנת נשמרת במסד הנתונים האינטרנטי.



איור 22-10. שימוש בלוקים של GotValue כדי לעבד את הנתונים המוחזרים ממסד הנתונים ולבצע פעולות שונות על סמך מה שמוחזר

ה- else של החסימה if else מופעל אם רשימה לא הוחזרה מהאינטרנט

מאגר מידע; זה קורה אם אף אחד לא השתמש באפליקציה עדיין. במקרה זה, נוצרת רשימת בוחרים חדשה עם כתובת האימייל של המשתמש הנוכחי בתור הפריט הראשון. הבוחר בסעיף אחד הזה לאחר מכן, הרשימה מאוחסנת במסד הנתונים האינטרנטי (בתקווה שגם אחרים יצטרפו!).

בקשת נתונים עם תגים שונים

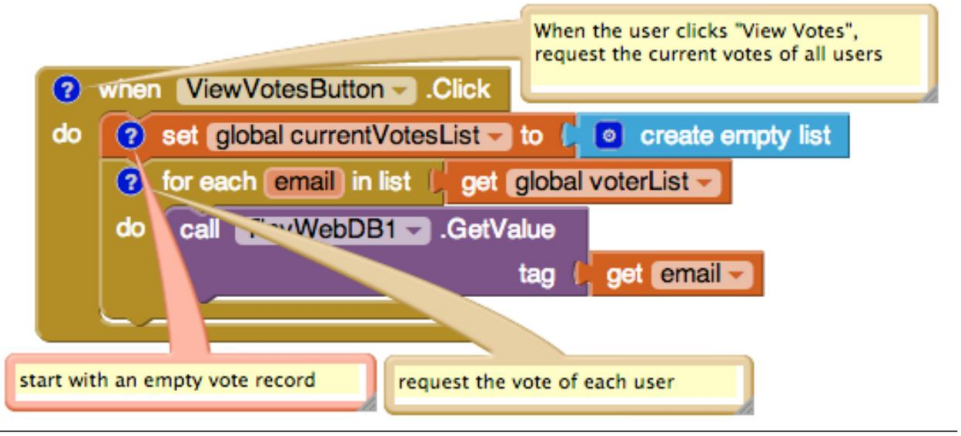
אפליקציית ההצבעה מנהלת עד כה רשימה של משתמשי אפליקציה. כל אדם יכול לראות את כתובות האימייל של כל שאר המשתמשים, אבל עדיין לא יצרנו חסימות לאחזור והצגת הצבעה של כל משתמש.

נזכיר שאירוע VoteButton.Click הגיש הצבעה עם צמד תג-ערך של הטופס "אימייל: הצבע". אם שני אנשים היו משתמשים באפליקציה ומצביעים, ערכי מסד הנתונים הרלוונטיים היו נראים בערך כמו טבלה 22-3.

טבלה .22-3צמדי התג-ערך המאוחסנים במסד הנתונים

תָג	ערך	
רשימת הבוחרים	[bill@zmail.com, joe@zmail.com]	
נקניקיותbill@zmail.com		
פיצהjoe@zmail.com		

כאשר המשתמש לוחץ על כפתור , ViewVotesהאפליקציה אמורה לאחזר את כל ההצבעות ממסד הנתונים ולהציג אותן. נניח שרשימת הבוחר כבר אוחזרה לרשימת הבוחרים המשתנה ; אנו יכולים להשתמש ב-a עבור כל אחד כדי לבקש את ההצבעה של כל אדם ברשימה, כפי שמוצג באיור .22-10



איור .22-11שימוש ב-a עבור כל בלוק כדי לבקש את ההצבעה של כל אדם ברשימה

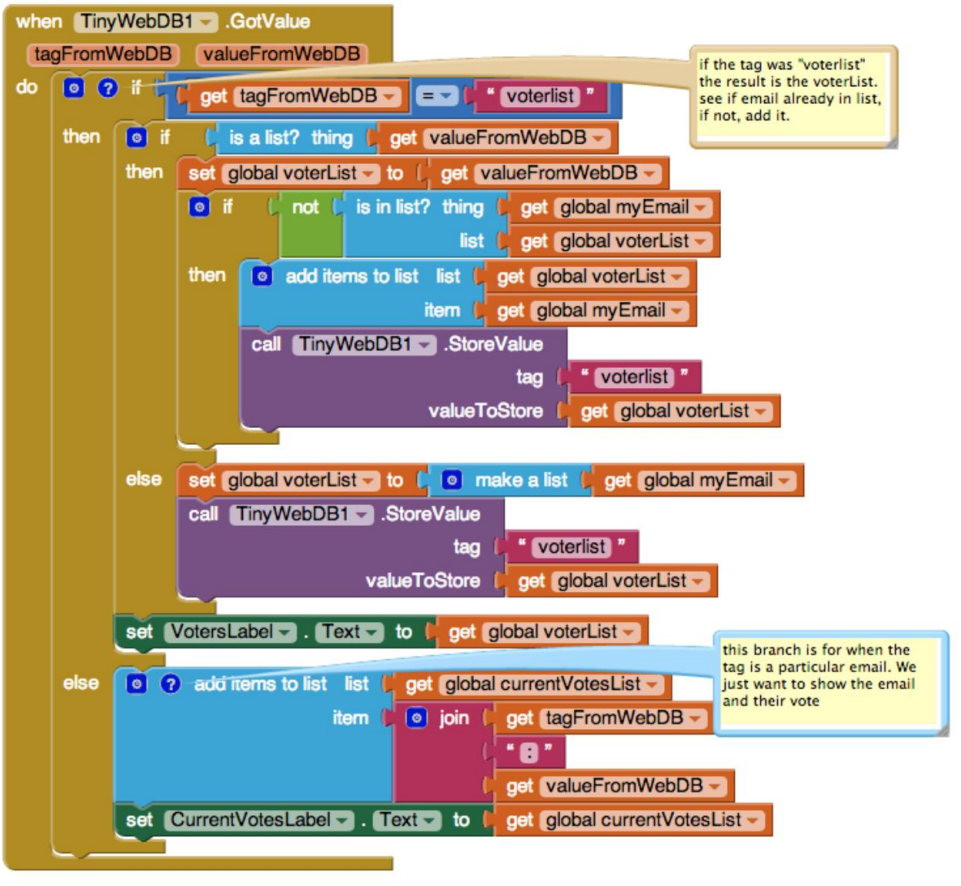
כאן אנו מאתחלים משתנה, currentVotesListלרשימה ריקה, כי המטרה שלנו היא כדי להוסיף את ההצבעות המעודכנות מהמאגר לרשימה זו. לאחר מכן, אנו משתמשים עבור כל אחד להתקשר ל- TinyWebDB1.GetValueעבור כל כתובת דוא"ל ברשימה, שולחים את הפריט הנוכחי של עבור כל אחד, ששמו שונה ל"אימייל", כתגית הבקשה. שימו לב שההצבעות לא יתווספו למעשה ל- currentVotesListעד שיגיעו דרך סדרה של GotValue אירועים.

כעת, כשאנחנו רוצים להציג את ההצבעות באפליקציה שלנו, הדברים הופכים קצת יותר מסובכים שוב. עם הבקשות מ- ViewVotesButton, TinyWebDB.GotValueתחזיר כעת נתונים הקשורים לכל תגי האימייל, כמו גם תג "רשימת הבוחרים" המשמש לאחזור רשימת כתובות הדוא"ל של המשתמש. כאשר האפליקציה שלך מבקשת יותר מפריט אחד ממסד הנתונים עם תגים שונים, עליך לקודד את TinyWebDB.GotValueכדי לטפל בכל הבקשות האפשריות. (אתה עשוי לחשוב שאתה יכול לנסות לקודד מרובים

מטפלי אירועים של GotValue , אחד עבור כל בקשה למסד נתונים -האם אתה יכול להבין מדוע זה לא יעבוד?)

כדי לטפל במורכבות הזו, למטפל האירועים GotValue יש ארגומנט tagFromWebDB שמודיע לך איזו בקשה הגיעה זה עתה. במקרה זה, אם התג הוא "רשימת בוחרים", עלינו להמשיך לעבד את הבקשה כפי שעשינו בעבר. אם התג הוא משהו אחר, אנו יכולים להניח שזהו האימייל של מישהו ברשימת המשתמשים, הנובע מהבקשות שהופעלו במטפל האירועים ViewVotesButton.Click . כאשר בקשות אלו מגיעות, אנו רוצים להוסיף את הנתונים הנכנסים -הבוחר וההצבעה -לרשימת ההצבעות הנוכחית כדי שנוכל להציג אותם למשתמש.

איור 22-11 מציג את כל מטפל האירועים TinyWebDB.GotValue .



איור 22-12. המטפל באירועים TinyWebDB.GotValue

הקמת מסד נתונים אינטרנטי

כפי שהזכרנו קודם לכן בפרק, מסד הנתונים האינטרנטי המוגדר כברירת מחדל בכתובת http://appinvtinywebdb.appspot.com מיועד למטרות אב טיפוס ובדיקות בלבד. לפני שאתה פורס אפליקציה עם משתמשים אמיתיים, עליך ליצור מסד נתונים ספציפי עבור האפליקציה שלך.

אתה יכול ליצור מסד נתונים אינטרנטי באמצעות ההוראות בכתובת http://appinventorapi.com/create-a-web-database-python-2-7. אתר זה הוקם על ידי אחד המחברים (Wolber) ומכיל קוד לדוגמה והנחיות להגדרת מסדי נתונים וממשקי API של App Inventor. ההוראות מכוונות אותך לאיזה קוד שתוכל להוריד ולהשתמש בו רק בשינוי מינורי של תצורה. הורדת הקוד זהה לזו המשמשת עבור מסד הנתונים האינטרנטי המוגדר כברירת מחדל שהוגדר על ידי App Inventor. הוא פועל על App Engine של גוגל, שירות מחשוב ענן שיארח את מסד הנתונים האינטרנטי שלכם בשרתי גוגל בחינם (טוב, לפחות עד שהאתר יקבל מספר מסוים של כניסות). על ידי ביצוע ההוראות, תוכל להפעיל מסד נתונים אינטרנט פרטי משלך התואם לפרוטוקולים של App Inventor תוך דקות ולהתחיל ליצור אפליקציות סלולריות התומכות באינטרנט המשתמשות בו.

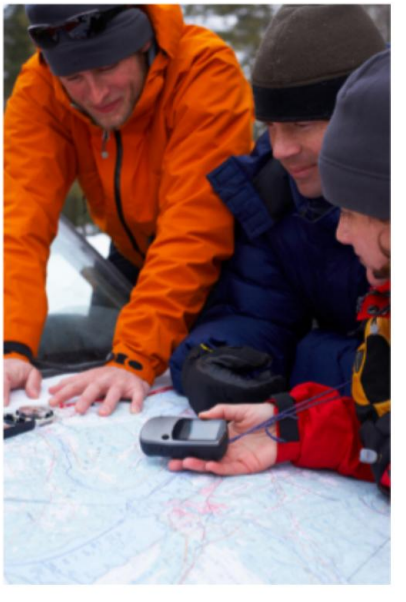
כאשר אתה יוצר ופרוס מסד נתונים אינטרנט מותאם אישית משלך, הכלי App Engine מספק לך כתובת URL שבה השרת שלך שוכן. אתה יכול לכוון את האפליקציה שלך להשתמש בשרת מסד הנתונים המותאם אישית שלך במקום ברירת המחדל appinvtinywebdb.appspot.com, על ידי שינוי המאפיין ServiceURL ברכיב TinyWebDB. לאחר שינוי המאפיין הזה, כל הקריאות ל- TinyWebDB.StoreValue ו- TinyWebDB.GetValue יתממשקו עם שירות האינטרנט החדש.

סיכום

App Inventor מקל על אחסון נתונים בהתמדה באמצעות רכיבי TinyDB ו- TinyWebDB שלו. הנתונים מאוחסנים תמיד כזוג תג-ערך, כאשר התג מזהה את הנתונים לצורך אחזור מאוחר יותר. השתמש ב-TinyDB כאשר מתאים לאחסן נתונים ישירות במכשיר. כאשר יש צורך לשתף נתונים בין טלפונים (למשל, עבור משחק מרובה משתתפים או אפליקציית הצבעה), תצטרך להשתמש ב-TinyWebDB, במקום זאת. TinyWebDB מסובך יותר מכיוון שאתה צריך להגדיר הליך התקשרות חוזר (המטפל באירועים GotValue) כמו גם שירות מסד נתונים אינטרנטי.

קריאה ותגובה לחיישנים

איור 23-1.



כוון את הטלפון שלך לשמיים, Google Sky Map-ו יגיד לך באילו כוכבים אתה מסתכל. הטה את הטלפון שלך, ותוכל לשלוט במשחק שאתה משחק. קח את הטלפון שלך לריצה היומית שלך, ואפליקציה מתעדת את המסלול שלך. כל האפליקציות הללו אפשריות מכיוון שלמכשירים הניידים שאנו נושאים יש חיישני היי-טק לזיהוי המיקום, הכיוון וההאצה שלנו.

בפרק זה, תחקור את רכיבי ה-AppInventor LocationSensor, OrientationSensor ו- AccelerometerSensor. לאורך הדרך, תלמדו על מערכת המיקום הגלובלית (GPS); מדדי התמצאות כגון גובה גובה, גלגול ואזימוט; וקצת מתמטיקה לעיבוד קריאות מד תאוצה.

יצירת מודעות למיקום אפליקציות

עד לפופולאריזציה של הטלפון החכם, המחשוב היה בנעילת שולחן העבודה. כן, מחשבים ניידים הם ניידים, אבל לא באותו מובן כמו המכשירים הזעירים שאנו סוחבים כעת בכיסים שלנו. המחשוב עזב את המעבדה ואת המשרד, וכעת הוא מתרחש בעולם, מעבר למגבלות של ארבעה קירות.

אפקט אחד משמעותי של נשיאת המחשוב שלנו איתנו הוא חדש, מעניין מאוד פיסת נתונים עבור כל אפליקציה: מיקום נוכחי. לדעת היכן אנשים נמצאים בזמן שהם מסתובבים בעולם יש השלכות מרחיקות לכת ופוטנציאל לעזור לנו מאוד בחיינו. יש לזה גם פוטנציאל לפלוש לפרטיות שלנו ולהוות פגיעה באנושות.

האנדרואיד, איפה המכונית שלי? אפליקציה (פרק 7) היא דוגמה לאפליקציה מודעת למיקום המספקת הטבה אישית. זה מאפשר לך לזכור מיקום קודם כדי שתוכל לחזור אליו במועד מאוחר יותר. האפליקציה הזו פרטית, כלומר פרטי המיקום שלך מאוחסנים רק במסד הנתונים של המכשיר שלך.

קבוצות יכולות גם להשתמש בחיישת מיקום. לדוגמה, קבוצת מטיילים עשויה לרצות עקבו אחר מקום הימצאו של זה במדבר, או שקבוצה של שותפים עסקיים עשויים לרצות למצוא אחד את השני בכנס גדול (או בר). יש אנשים שמשתמשים באפליקציות "צ'ק-אין" כאלה מדי יום.

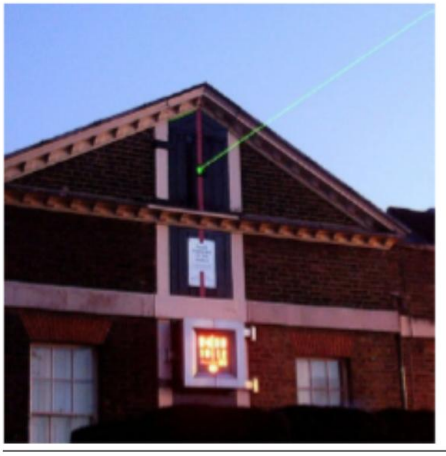
סוג אחר של אפליקציה מודעות למיקום משתמש בכלי מציאות רבודה. אפליקציות אלה משתמשות המיקום שלך והכיוון של הטלפון כדי לספק מידע שכבת-על שמגביר את ההגדרה הטבעית. אז אולי תכוון טלפון לבניין ותראה את המחיר שלו בשוק הנדל"ן, או שאתה יכול ללכת ליד צמח אקזוטי בגן בוטני ואפליקציה יכולה לספר לך על המינים שלו.

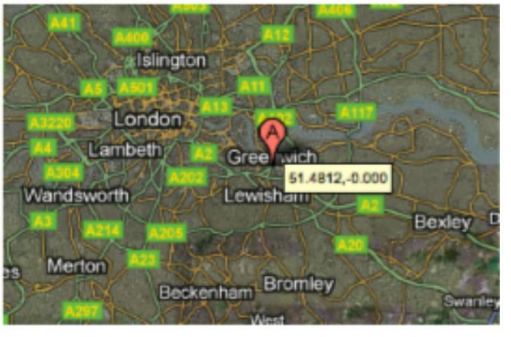
מערכת המיקום הגלובלית

כדי ליצור אפליקציה מודעת למיקום, תחילה עליך להבין כיצד פועלת מערכת המיקום הגלובלית .(GPS)נתוני GPSנוצרים באמצעות סדרה של לוויינים גיאו-סינכרוניים המתוחזקים על ידי ממשלת ארצות הברית. כל עוד יש לך קו ראייה ללא הפרעה לשלושה לוויינים לפחות במערכת, הטלפון שלך יכול לקבל קריאה. קריאת GPSמורכבת מקו הרוחב, קו האורך והגובה שלך. קו הרוחב הוא המרחק צפונה או דרומה אתה ביחס לקו המשווה, כאשר ערכי צפון חיוביים ודרום שליליים. הטווח הוא 90- עד .90איור 1-23מציג מפת גוגל של נקודה ליד קיטו, אקוודור. קו הרוחב המוצג במפה הוא ,0.01- רק בקושי מדרום לקו המשווה!

איור .23-2קיטו, אקוודור, נמצאת על קו המשווה

קו אורך הוא כמה רחוק אתה ממזרח או מערב מהמרידיאן הראשי; לקואורדינטות מזרח יש ערכים חיוביים וקואורדינטות מערביות הן שליליות. פריים מרידיאן, המשתרע מצפון לדרום, ממוקם בגריניץ', עיירה ליד לונדון המהווה את ביתו של מצפה הכוכבים המלכותי, הארגון שבמקור הקים את פריים מרידיאן כבסיס למדידה לאסטרונומים ולנווטים כאחד. המפה באיור 23-2מציגה את גריניץ' וקו האורך שלה .0.0





איור .23-3מצפה הכוכבים המלכותי בגריניץ' יורה קרן אור לאורך מרידיאן הפריים

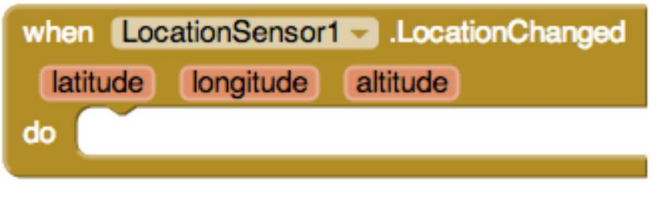
ערכי קו האורך נעים בין 180-ל-081. איור 23-3מציג נקודה ברוסיה, מאוד קרוב לאלסקה, שיש לה קו אורך .180.0אפשר לומר שמיקום כזה נמצא באמצע העולם מגריניץ' (קו אורך .(0.0

איור 23-4. לנקודה ליד הגבול בין רוסיה לאלסקה יש קו אורך 180

חישת מיקום עם ממציא אפליקציה

App Inventor מספק את רכיב ה- LocationSensor לגישה למידע GPS. לרכיב יש מאפיינים עבור קו רוחב, קו אורך וגובה . זה גם מתקשר עם מפות Google, כך שתוכל לקבל קריאה עבור כתובת הרחוב הנוכחית שלך.

LocationSensor.LocationChanged, בתמונה באיור 23-4, הוא המטפל באירועי מפתח עבור ה- LocationSensor.

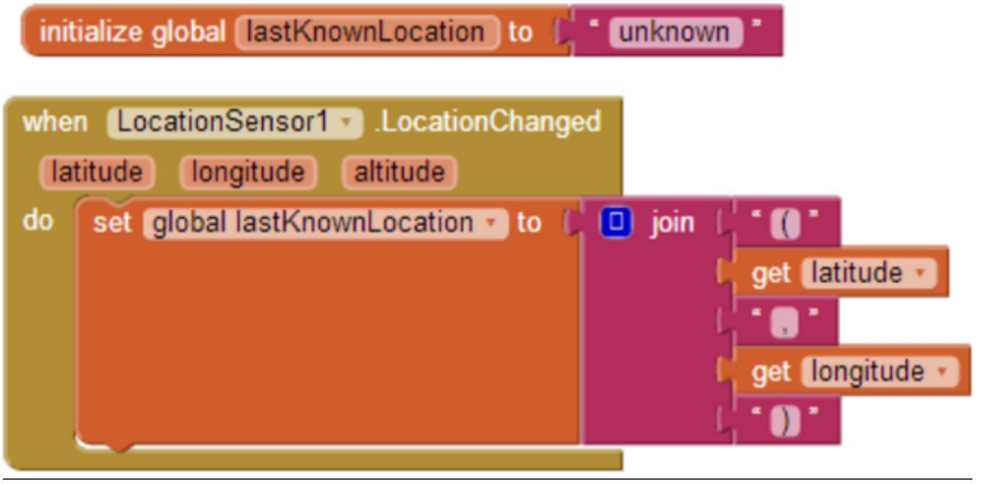


איור 23-5. המטפל באירועים LocationSensor1.LocationChanged

אירוע זה מופעל בפעם הראשונה שהחיישן יוצר קריאה ובכל פעם אחר כך שהטלפון מזיז מספיק כדי שמידע חדש ייקרא. לעתים קרובות יש עיכוב של לא מעט שניות לפני הקריאה הראשונה של אפליקציה, ולפעמים המכשיר לא מצליח לקבל קריאה כלל. לדוגמה, אם אתה בתוך הבית ולא מחובר ל-iFiW, ייתכן שהמכשיר לא יקבל קריאה. לטלפון שלך יש גם הגדרות שבאמצעותן תוכל להפעיל קריאת GPS כדי לחסוך בחיי סוללה; זוהי סיבה אפשרית נוספת לכך שהרכיב לא יכול לקבל קריאה. מסיבות אלו, אל תניח שה- LocationSensor

לנכסים יש הגדרה חוקית עד לאירוע LocationSensor.LocationChanged
מתרחשת.

אחת הדרכים להתמודד עם הלא ידועים בחיישת מיקום היא ליצור משתנה
,lastKnownLocationאתחל אותו ,"unknown"-לולאחר מכן בקש מהמטפל באירועים
LocationSensor.LocationChangedלשנות את הערך של המשתנה הזה, כפי שמוצג באיור
23-5.



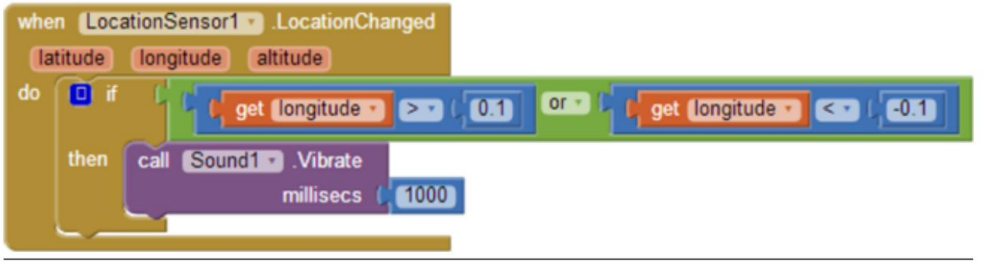
איור .23-6הערך של המשתנה lastKnownLocationמשתנה בכל פעם שהמיקום משתנה

על ידי תכנות המטפל באירועים LocationSensor.LocationChangedבאופן זה, אתה
תמיד יכול להציג את המיקום הנוכחי או להקליט אותו במסד נתונים, כאשר "לא ידוע" מופיע עד
הקריאה הראשונה. אסטרטגיה זו משמשת בתוכנית ללא הודעות טקסט בזמן נהיגה! אפליקציה
(פרק ;(4האפליקציה הזו מגיבה אוטומטית להודעות SMSוכוללת "לא ידוע" או את הקריאה
האחרונה שנלקחה בתגובה.
אתה יכול גם לשאול במפורש אם לחיישן יש קריאה באמצעות
בלוק LocationSensor.HasLongitudeLatitudeמאויר באיור .23-6

איור .23-7בדיקה אם לחיישן יש קריאה באמצעות בלוק
HasLongitudeLatitude

בדיקת גבולות

שימוש נפוץ אחד באירוע LocationChangedהוא לבדוק אם המכשיר נמצא בתוך גבול, או אזור מוגדר. לדוגמה, קחו בחשבון את הקוד באיור 23-7, המרעיד את הטלפון בכל פעם שקריאה חדשה מראה שאדם התרחק מ-1.0 מעלות קו אורך מהפריים מרידיאן.



איור 23-8. קריאה אינה קרובה לפריים מרידיאן, הטלפון רוטט

לבדיקת גבולות כזו יש יישומים רבים; למשל, אזהרת שחרורים על תנאי אם הם מתקרבים למרחק שנקבע בחוק מביתם, או התראה להורים או למורים אם ילד עוזב את אזור מגרש המשחקים. אם תרצה לראות דוגמה קצת יותר מורכבת, ראה את הדיון בפרק 18על בלוקים מותנים.

ספקי מידע על מיקום: GPS, WIFIומזהה סלולרי

מכשיר אנדרואיד יכול לקבוע את המיקום שלו במספר דרכים. השיטה המדויקת ביותר -בטווח של מטרים ספורים -היא באמצעות לווייני ה-SPG. עם זאת, לא תקבל קריאה אם אתה בפנים או שיש גורדי שחקים או מכשולים אחרים סביבך; אתה צריך נתיב פנוי לפחות לשלושה לווויינים במערכת.

אם ה-SPG אינו זמין או שהמשתמש השבית אותו, המכשיר יכול לקבל את מיקומו דרך רשת אלחוטית. אתה חייב להיות ליד נתב WiFi, כמובן, וקריאת המיקום שתקבל הוא קו הרוחב/קו האורך של אותה תחנת WiFi

דרך שלישית שבה מכשיר יכול לקבוע מיקום היא באמצעות מזהה תא. מזהה סלולרי מספק א מיקום הטלפון בהתבסס על עוצמת האותות ממגדלי הטלפון הסלולרי הסמוכים. זה בדרך כלל לא מאוד מדויק אלא אם כן יש לך מגדלים סלולריים רבים בקרבתך. עם זאת, הוא אכן משתמש בכמות הנמוכה ביותר של סוללה בהשוואה לקישוריות GPSאו WiFi.

שימוש בחיישן הכיוון

אתה יכול להשתמש ב- OrientationSensorעבור אפליקציות דמויות משחק בהן המשתמש שולט בפעולה על ידי הטיית המכשיר. זה יכול לשמש גם כמצפן כדי לגלות לאיזה כיוון (צפון/דרום, מזרח/מערב) הטלפון מכוון.

ל- OrientationSensorיש חמישה מאפיינים, שכולם אינם מוכרים לרוב האנשים מלבד מהנדסי אווירונאוטיקה:

גלגול (שמאל-ימין)

הגלגול הוא 0מעלות כשהמכשיר מיושר, עולה ל-09 מעלות כשהמכשיר מוטה לכיוון הצד השמאלי, ויורד ל-09 מעלות כשהמכשיר מוטה לכיוון הצד הימני.

גובה גובה (מעלה-גב)

גובה הצליל הוא 0מעלות כשהמכשיר מיושר, עולה ל-09 מעלות כשהמכשיר מוטה כך שהחלק העליון שלו מצביע כלפי מטה, ועולה עוד יותר ל-081 מעלות כשהמכשיר מתהפך. באופן דומה, כאשר המכשיר מוטה כך שהחלק התחתון שלו מצביע כלפי מטה, ה-hctiP פוחת ל-09 מעלות ולאחר מכן מטה ל-081 מעלות כאשר הוא מופנה עד הסוף.
על.

אזימוט (מצפן)

אזימוט הוא 0מעלות כאשר החלק העליון של המכשיר מצביע לצפון, 90מעלות כאשר הוא מצביע מזרחה, 180מעלות כאשר הוא מצביע לדרום, ו-072 מעלות כאשר הוא מצביע מערבה.

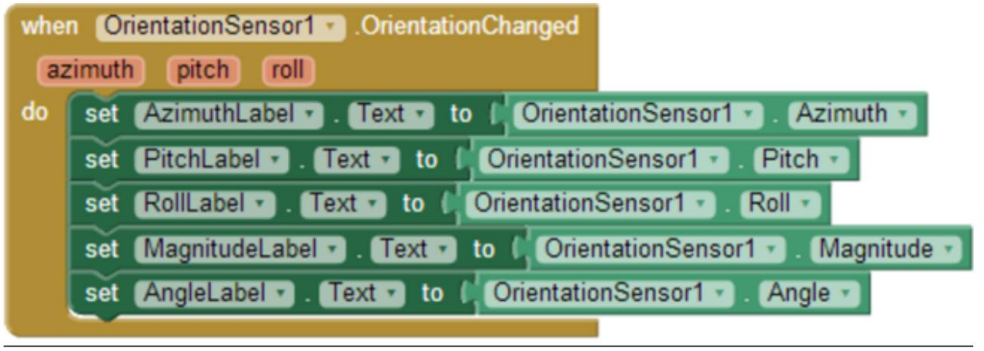
גודל (מהירות של כדור מתגלגל)

גודל מחזיר מספר בין 0ל-1 המציין עד כמה המכשיר מוטה. ערכו מציין את הכוח שמפעיל כדור שמתגלגל על פני המכשיר.

זווית (זווית של כדור מתגלגל)

זווית מחזירה את הכיוון בו מרצפת המכשיר. כלומר, הוא מציין את כיוון הכוח שיופעל על ידי כדור שמתגלגל על פני המכשיר.

OrientationSensorמספק את אירוע , OrientationChangedהמופעל בכל פעם שהכיוון משתנה. כדי לחקור את המאפיינים הללו יותר, בואו נכתוב אפליקציה שממחישה כיצד המאפיינים משתנים כשהמשתמש מטה את המכשיר. פשוט הוסף חמש תוויות כותרות וחמש תוויות אחרות כדי להציג את הערכים הנוכחיים של המאפיינים ברשימה הקודמת. לאחר מכן, הוסף את הבלוקים המוצגים באיור .8-23



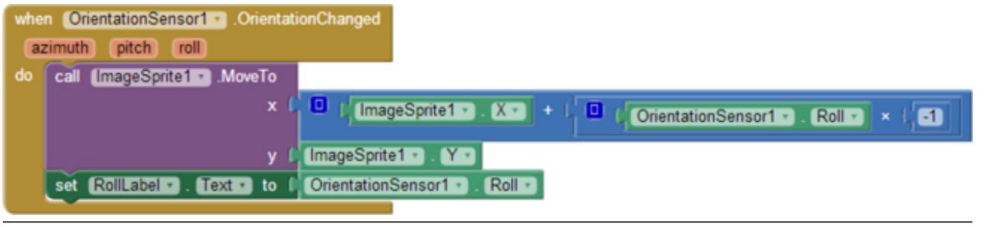
איור 23-9. חסימות להצגת נתוני OrientationSensor

שימוש בפרמטר הגלגול כדי להזיז אובייקט

הפעם, בואו ננסה להזיז תמונה שמאלה או ימינה על המסך בהתבסס על הטיית המשתמש, כפי שאתם עשויים לעשות במשחק יריות או נהיגה. גרור החוצה קנבס והגדר את הרוחב ל"מלא הורה" ואת הגובה ל-002 פיקסלים. לאחר מכן, הוסף ImageSprite או Ball בתוך הקנבס, והוסף תחתיו תווית בשם RollLabel כדי להציג ערך מאפיין, כפי שמוצג באיור 23-9.

איור 23-10. ממשק משתמש לחקירה כיצד ניתן להשתמש בגלגול כדי להזיז תמונה

מאפיין ה- Roll של OrientationSensor יציין אם הטלפון מוטה שמאלה או ימינה -אם תחזיק את הטלפון זקוף ותטה אותו מעט שמאלה, תקבל קריאה חיובית עבור roll; -האם תטה אותו מעט ימינה, תקבל קריאה שלילית. לכן, אתה יכול לאפשר למשתמש להזיז אובייקט עם מטפל באירועים כמו זה שמוצג באיור 23-10.



איור .23-11תגובה לשינויים במאפיין Rollעם האירוע OrientationChanged

הבלוקים מכפילים את הגלילה ב-1, מכיוון שהטיה שמאלה נותנת גלגול חיובי וצריכה להזיז את האובייקט שמאלה (ובכך להקטין את קואורדינטת ה-x). לסקירה של אופן הפעולה של מערכת הקואורדינטות באפליקציות מונפשות, ראה פרק 17.

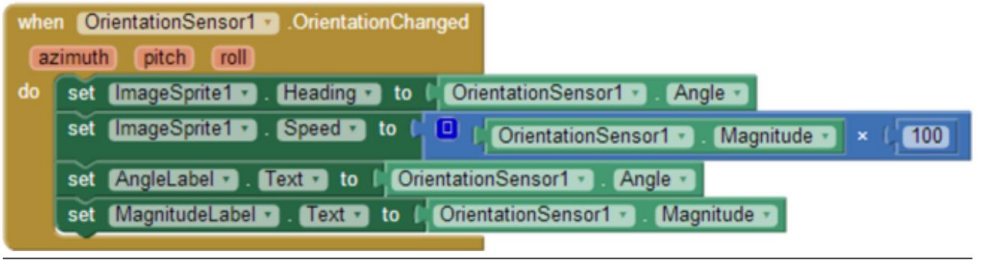
שימו לב שהאפליקציה הזו פועלת רק כשהמכשיר במצב פורטרט (זקוף), לא במצב נוף. כפי שהוא, אם תטה את הטלפון יותר מדי, המסך ישתנה למצב נוף והתמונה תישאר מרוחקת בצד שמאל של המסך. הסיבה היא שאם המכשיר נמצא על הצד, הוא מוטה שמאלה וכך תמיד יקבל קריאה חיובית לגלגול. קריאת גליל חיובית, כפי שמוצגת בבלוקים באיור 23-10, תמיד תקטין את קואורדינטת ה-x.

שימו לב ש-AppInventor מספק את המאפיין Screen.ScreenOrientation, שבו אתה יכול להשתמש כדי לנעול את הכיוון אם אתה לא רוצה שהוא יעבור בין המצבים.

תנועה לכל כיוון על ידי שימוש בכותרת ובגודל

הדוגמה בסעיף הקודם מזיזה את התמונה שמאלה או ימינה. אם אתה רוצה לאפשר תנועה בכל כיוון, אתה יכול להשתמש במאפייני הזווית והגודל של ה- OrientationSensor. אלו הם המאפיינים המשמשים להזזת פרת משה רבנו במשחק המתואר בפרק 5.

באיור 23-11, ניתן לראות את הבלוקים של אפליקציית בדיקה שבה המשתמש מטה את המכשיר כדי להזיז דמות לכל כיוון (צריך שתי תוויות וספרייט תמונה לדוגמא זו).



איור .23-12הזזת דמות באמצעות זווית וגודל

נסה את זה. המאפיין Magnitude, מציין כיצד ערך בין 0ל-1, במידה רבה המכשיר מוטה. באפליקציית בדיקה זו, התמונה זזה מהר יותר ככל שערך הגודל עולה.

שימוש בטלפון כמצפן

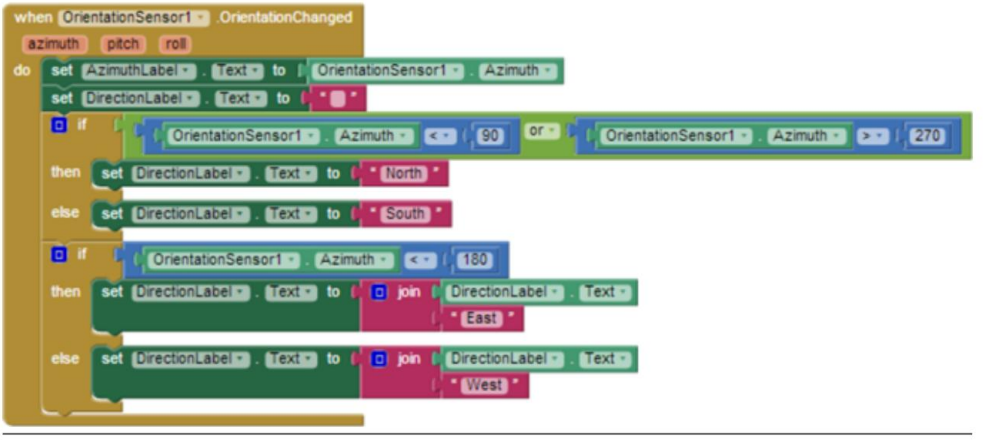
אפליקציות ואפליקציות מצפן כמו Google Sky Map צריכות לדעת את כיוון הטלפון בעולם, מזרח/מערב וצפון/דרום. מפת השמיים משתמשת במידע כדי לכסות מידע על קבוצות הכוכבים שאליהן הטלפון מצביע.

קריאת אזימוט שימושית עבור סוג זה של אוריינטציה . אזימוט הוא תמיד בין 0ל-063 מעלות, כאשר 0הוא צפון; ,90מזרח; ,180דרום; ו-072, מערב.

לפיכך, קריאה של 45פירושה שהטלפון מצביע לצפון מזרח, 135אומר דרום מזרח, 225אומר דרום מערב ו-513 אומר צפון מערב.

הבלוקים באיור 23-12מיועדים למצפן פשוט שמציג בטקסט אשר לכיוון שאליו הטלפון מצביע (למשל, צפון מערב).

כפי שאולי שמתם לב, הבלוקים מציגים רק אחת מארבע אפשרויות: צפון מערב, צפון מזרח, דרום מערב ודרום מזרח. כאתגר, בדוק אם אתה יכול לשנות אותו כך שיראה רק כיוון בודד (צפון, דרום, מזרח או מערב) אם הקריאה מציינת שאתה מצביע בטווח של כמה מעלות ממנו.



איור 23-13. תכנות מצפן פשוט

שימוש במד התאוצה

תאוצה היא קצב השינוי של המהירות לאורך זמן. אם אתה לוחץ את הרגל על דוושת הדלק של המכונית שלך, המכונית מאיץ -מהירותה עולה בקצב מסוים.

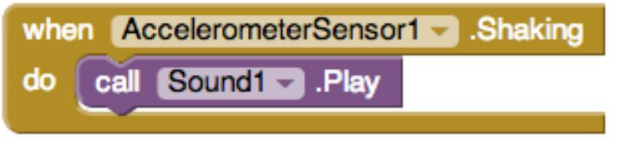
מד תאוצה כמו זה שבמכשיר האנדרואיד שלך מודד תאוצה, אבל מסגרת ההתייחסות שלו היא לא המכשיר במנוחה, אלא המכשיר בנפילה חופשית: אם אתה

זרוק את הטלפון, הוא ירשום קריאת תאוצה של .0במילים פשוטות, הקריאות לוקחות בחשבון את כוח המשיכה.

אם אתה רוצה לדעת יותר על הפיזיקה של העניין, תצטרך להתייעץ הספרים שלך הקשורים לאיינשטיין. אבל בסעיף זה, נחקור את מד התאוצה מספיק כדי להתחיל. אפילו נבחן אפליקציה שיכולה לעזור להציל חיים!

תגובה לרעד של המכשיר

אם השלמת את אפליקציית Hello Purrבפרק ,1כבר השתמשת בחיישן האצה. באפליקציה הזו, השתמשת באירוע Accelerometer.Shakingכדי לגרום לקיטי מיאו כאשר הטלפון נער, כפי שמוצג באיור .23-13



איור .23-14השמעת צליל כשהטלפון מזועזע

שימוש בקריאות של חיישן התאוצה

כמו שאר החיישנים, למד התאוצה יש אירוע למועד שינוי הקריאות, AccelerometerSensor.AccelerationChanged.לאירוע הזה יש שלושה ארגומנטים התואמים לתאוצה בתלת מימד:

xAccel

חיובי כאשר המכשיר מוטה ימינה (כלומר, הצד השמאלי שלו מורם), ושלילי כאשר המכשיר מוטה שמאלה (הצד הימני שלו מורם).

yAccel

חיובי כאשר החלק התחתון של המכשיר מורם, ושלילי כאשר החלק העליון שלו מורם.

zAccel

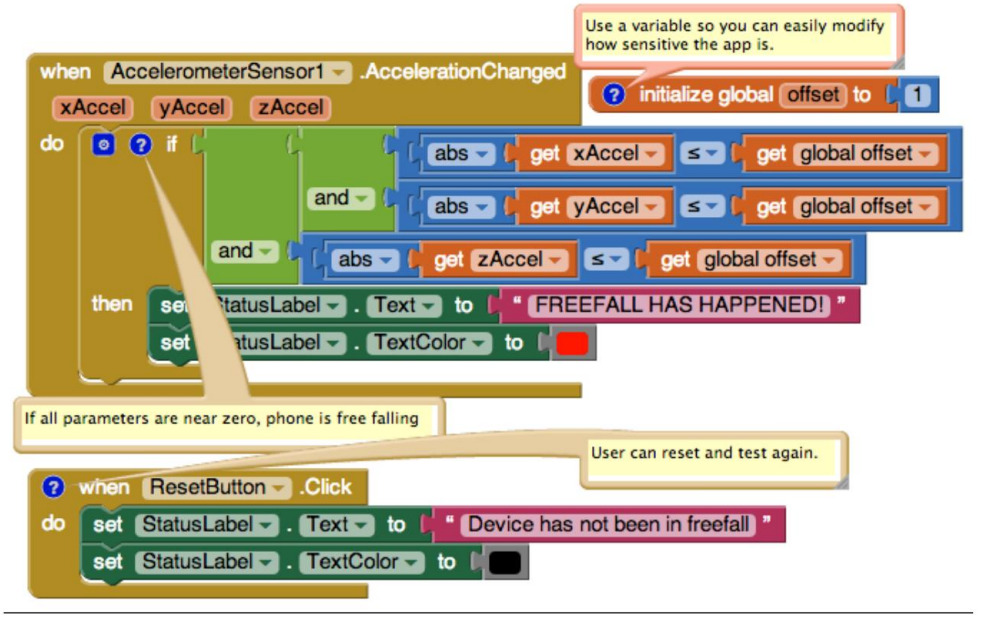
חיובי כאשר תצוגת המכשיר פונה כלפי מעלה, ושלילי כאשר התצוגה פונה כלפי מטה.

זיהוי נפילה חופשית

אנו יודעים שאם כל קריאות התאוצה קרובות ל-0, המכשיר נופל חופשי לקרקע. עם זאת בחשבון, אנו יכולים לזהות אירוע נפילה חופשית על ידי בדיקת הקריאות באירוע . AccelerometerSensor.AccelerationChangedאתה יכול להשתמש בלוקים כאלה, עם הרבה בדיקות, כדי לזהות מתי קשיש נפל ולשלוח אוטומטית הודעת SMSבתגובה.

שימוש במד התאוצה

איור 23-14מציג את החסימות של אפליקציה שפשוט מדווחת שיש לנפילה חופשית התרחש (ומאפשר למשתמש ללחוץ על כפתור איפוס כדי לבדוק שוב).1



איור 23-15. דיווח מתי התרחשה נפילה חופשית

בכל פעם שהחיישן מקבל קריאה, הבלוקים בודקים את מידות x, y, z-ראה אם הם קרובים ל-0 (אם הערך המוחלט שלהם קטן מ-1). אם שלושתם קרובים ל-0, האפליקציה משנה תווית סטטוס כדי לציין שהטלפון נמצא בנפילה חופשית. כאשר המשתמש מקיש על כפתור האיפוס, תווית הסטטוס מאופסת למצבה המקורי ("המכשיר לא היה בנפילה חופשית").

סיכום

חיישנים מעוררים עניין רב באפליקציות לנייד מכיוון שהם מאפשרים למשתמשים שלך ליצור אינטראקציה אמיתית עם הסביבה שלהם. על ידי שימוש במחשוב נייד, אתה פותח עולם שלם של הזדמנויות בחוויות משתמש ופיתוח אפליקציות. עם זאת, תצטרך לחשוב היטב כיצד, היכן ומתי אתה משתמש בחיישנים באפליקציות שלך. לאנשים רבים יש חששות בפרטיות, וייתכן שהם לא ישתמשו באפליקציה שלך אם הם מודאגים לגבי מה שאתה עושה עם נתוני החיישנים שלהם. עוד,

1אתה יכול ללחוץ לחיצה ימנית על בלוק ולבחור "קלטות מוטבעות" כדי לשנות את האופן שבו בלוקים מופיעים. זה נעשה עבור הבלוקים בדוגמה זו כדי להקטין את הרוחב של המטפל באירועים.

עם כל האפשרויות במשחקים, רשתות חברתיות, נסיעות ועוד, האפשרויות להטמעות חיוביות הן כמעט אינסופיות.

תקשורת עם האינטרנט

איור .24-1

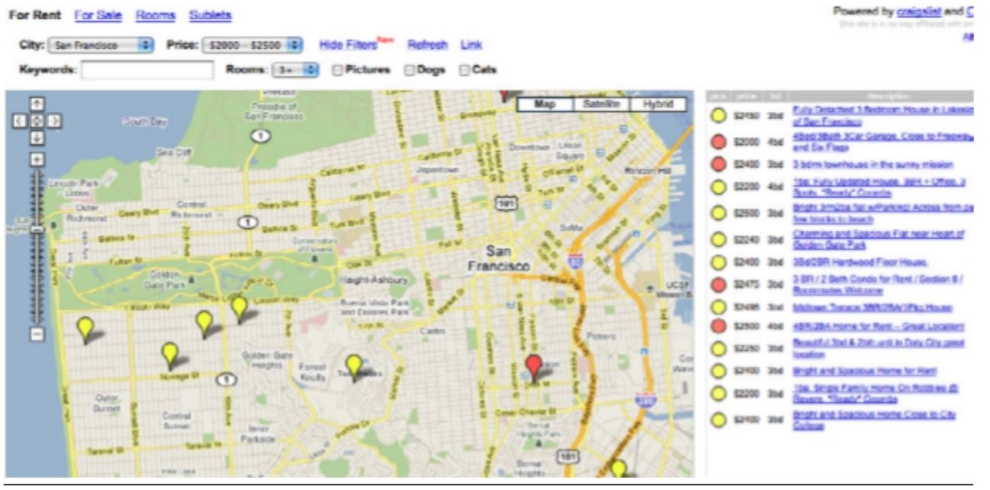
הטכנולוגיה הניידת והטבע הנפוצה של האינטרנט שינו את העולם בו אנו חיים. כעת אתה יכול לשבת בפארק ולעשות את הבנקאות שלך, לחפש Amazon.com-בכדי למצוא ביקורות על הספר שאתה קורא, ולבדוק בטוויטר כדי לראות מה אנשים בכל פארק אחר בעולם חושבים על.
טלפונים ניידים עברו הרבה מעבר להתקשרות והודעות טקסט -כעת, יש לך גם גישה מיידית לנתוני העולם.

אתה יכול להשתמש בדפדפן של הטלפון שלך כדי להגיע לאינטרנט, אבל לעתים קרובות המסך הקטן והמהירות המוגבלת של מכשיר נייד יכולים להפוך את זה לבעייתי.

אפליקציות מותאמות אישית, שתוכננו במיוחד למשוך גושים קטנים של מידע מתאים במיוחד מהרשת, יכולות לספק אלטרנטיבה אטרקטיבית יותר לדפדפן הנייד.

בפרק זה, נסקור את רכיבי App Inventor שניגשים למידע מהאינטרנט. תלמד כיצד להציג דף אינטרנט בממשק המשתמש של האפליקציה שלך, ותלמד על ממשקי API וכיצד לגשת למידע משירות אינטרנט.
יצירתיות היא לערבב מחדש את העולם, שילוב (מעיכה) של רעיונות ותכנים קיימים בדרכים חדשות מעניינות. אמינם הוא בין אמנים רבים בעשורים האחרונים שהפכו את המוזיקה לפופולרית כאשר הציב את הקול שלו Slim Shady על רצועות .Vanilla Ice-ו AC/DC סוג זה של "דגימה" נפוץ כיום, ואמנים רבים, כולל Negativland-ו Girl Talk מתמקדים בעיקר ביצירת רצועות חדשות משילוב של תוכן ישן.

עולם האינטרנט והמובייל אינם שונים: אתרים ואפליקציות לערבב מחדש תוכן מ מקורות נתונים שונים, ורוב האתרים מעוצבים כעת עם מחשבה על יכולת פעולה הדדית כזו. דוגמה להמחשה ל-puhsam באינטרנט היא Housing Maps, בתמונה באיור 24-1, שלוקחת מידע על השכרת דירה מ- Craigslist ומרסקת אותו עם API-השל מפות Google.

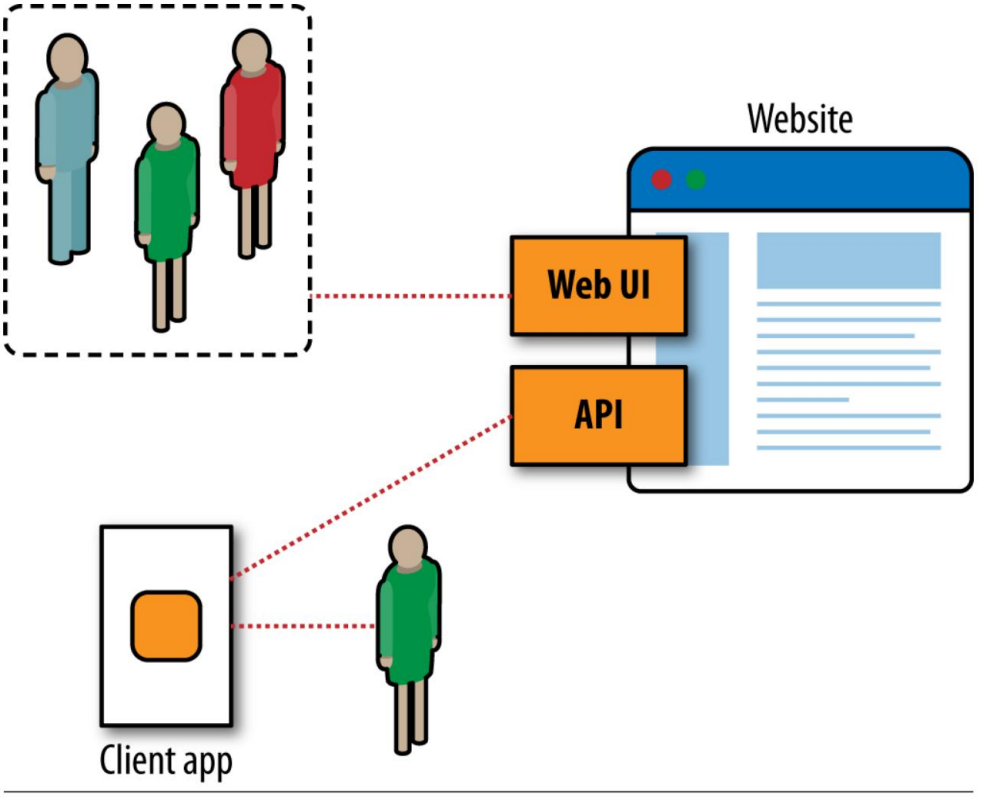


איור .2-24מפות דיור מעבדות מידע מ-CraigslistוGoogle Maps

אפשריות משימות הדומות למפות דיור בגלל שירותים כמו מפות גוגל לספק גם אתר וגם ממשק APIשל שירות אינטרנט מתאים. אנו, בני האדם, מבקרים ב-http://maps.google.com/בדפדפן , אך אפליקציות כגון Housing Mapsמתקשרות בין מכונה למכונה באמצעות ממשק API-השל מפות Google. Mashupsמעבדים את הנתונים, משלבים אותם עם נתונים מאתרים אחרים (למשל, Craigslist),ואז מציגים אותם בחדשים ומעניינים דרכים.

כמעט כל אתר פופולרי מספק כעת את החלופה הזו, גישה ממכונה למכונה. התוכנית המספקת את הנתונים נקראת שירות אינטרנט, והפרוטוקול לאופן שבו אפליקציית לקוח צריכה לתקשר עם השירות נקרא ממשק מתכנת יישומים, או .APIבפועל, המונח APIמשמש גם להתייחס לשירות האינטרנט.

שירות האינטרנט של אמזון (AWS)היה אחד משירותי האינטרנט הראשונים, שכן אמזון הבינה שפתיחת הנתונים שלה לשימוש על ידי ישויות צד שלישי תוביל בסופו של דבר למכירת ספרים נוספים. כשפייסבוק השיקה את API-השלה ב-7002, אנשים רבים הרימו גבות. הנתונים של פייסבוק הם לא פרסומות ספרים, אז למה שהיא תאפשר לאפליקציות אחרות "לגנוב" את הנתונים האלה ועלול למשוך משתמשים רבים מאתר פייסבוק (והפרסומות שלו!)? עם זאת, הפתיחות שלה הובילה את פייסבוק להפוך לפלטפורמה במקום רק לאתר -כלומר תוכניות אחרות יכולות להתבסס על הפונקציונליות של פייסבוק ולהיעזר בה, ואף אחד לא יכול להתווכח עם הצלחתה כיום. עד שהושק טוויטר ב-9002, גישה API-להייתה ציפייה, לא חידוש, וטוויטר פעלה בהתאם. כעת, כפי שמוצג באיור ,2-24רוב האתרים מציעים גם ממשק APIוגם ממשק אנושי.



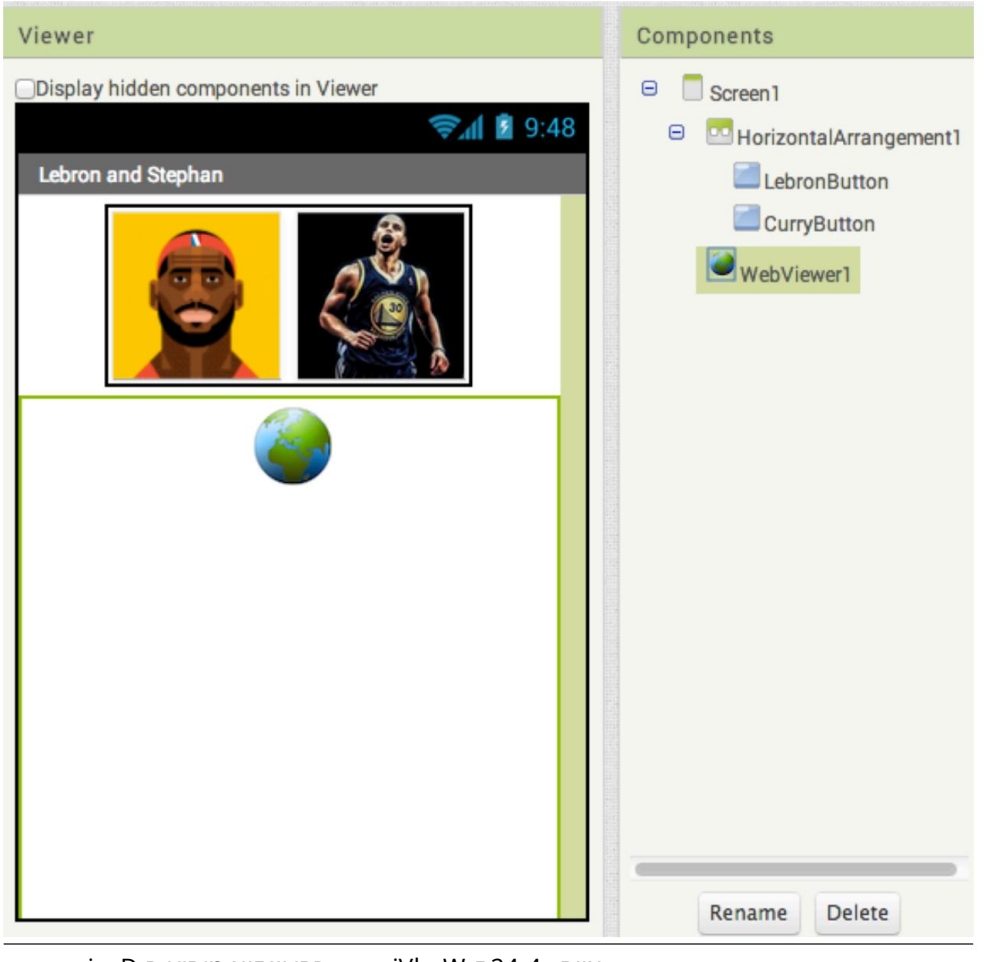
איור .3-24רוב האתרים מספקים גם ממשק אנושי וגם ממשק APIעבור אפליקציות לקוח

לפיכך, הרשת היא דבר אחד עבורנו בני האדם הממוצעים (אוסף של אתרים לביקור). ל מתכנתים, זהו מסד הנתונים הגדול והמגוון ביותר בעולם של מידע.

רכיב ה-reweiVbeW

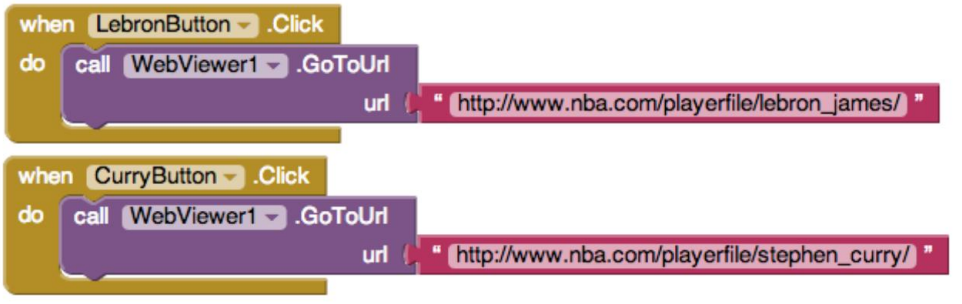
רכיב WebViewerמאפשר לך להציג דף אינטרנט בתוך האפליקציה שלך. אתה יכול להציג דף של מפות Googleמציג את מיקומו הנוכחי של המשתמש, דף טוויטר המציג את הנושאים הפופולריים ביותר הקשורים לאפליקציה שלך, או דף nba.com-מהמציג את הנתונים הסטטיסטיים של השחקנים האהובים עליך.

WebViewer(ראה איור 3-24דומה לרכיב Canvasבכך שהוא מגדיר תת-פאנל של המסך. אבל בעוד שקנבס משמש לציורים והנפשות, WebViewerמציג דף אינטרנט.



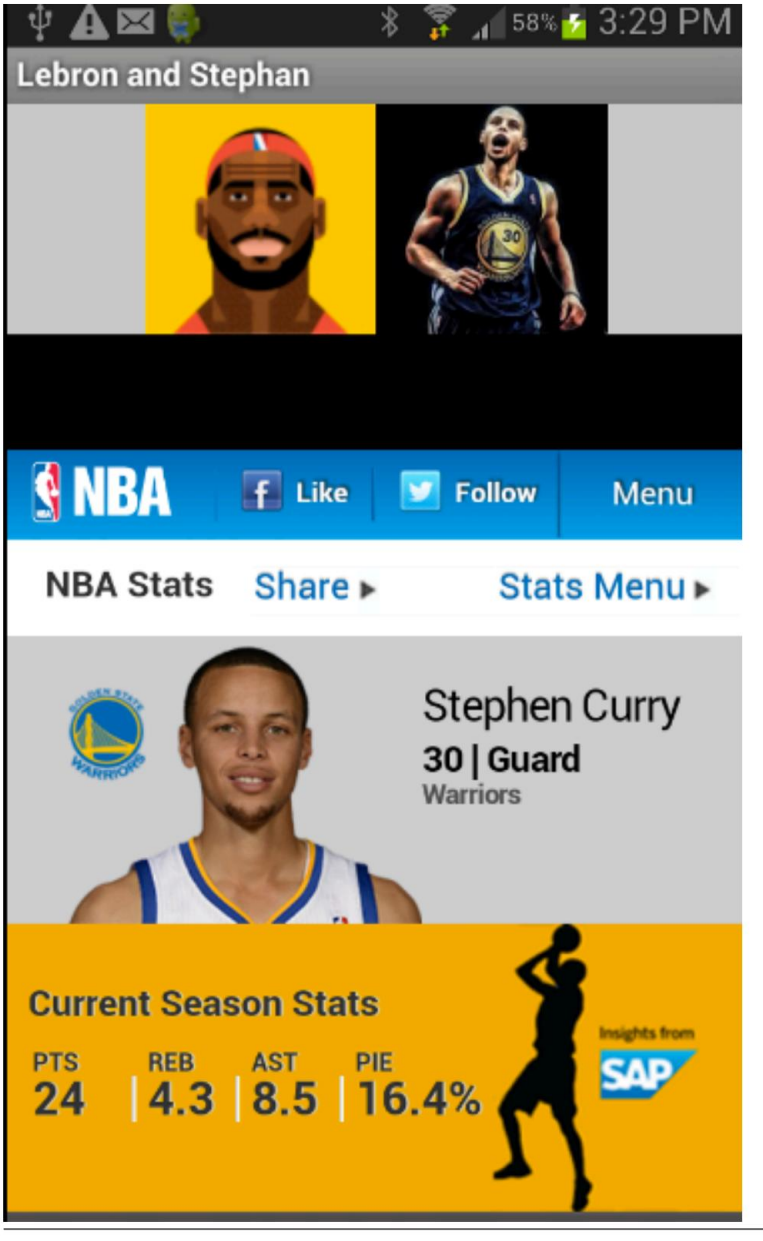
איור 24-4. reweiVbeW-ה כפי שהוא מופיע ב-rengiseD.

אתה יכול לגרור ב-reweiVbeW מהמגירה של ממשק המשתמש. לאחר מכן תוכל לשנות באופן דינמי את כתובת האתר המופיעה, כמו באיור 24-4, שמתארת בלוקים מאפליקציה המציגה את הנתונים הסטטיסטיים של שחקני ה-ABN לברון ג'יימס וסטיבן קארי:



איור 24-5. חסימות להצגת דף האינטרנט עבור השחקנים שנבחרו

אם המשתמש יקיש על התמונה של סטיבן קארי, האפליקציה תציג את הדף שלו מ-nba.com, כמו WebViewer, באיור 24-5.



איור 24-6. WebViewer באפליקציה

רכיב האינטרנט

בעוד ש-reweiVbeW מציג דף אינטרנט, רכיב האינטרנט , רכיב חדש יחסית ב-ppA Inventor, מאפשר לאפליקציה לתקשר עם שירות אינטרנט באמצעות פרוטוקול (HTTP) Hypertext Transfer-ההסטנדרטי. פרוטוקול זה מספק Get, Put ו

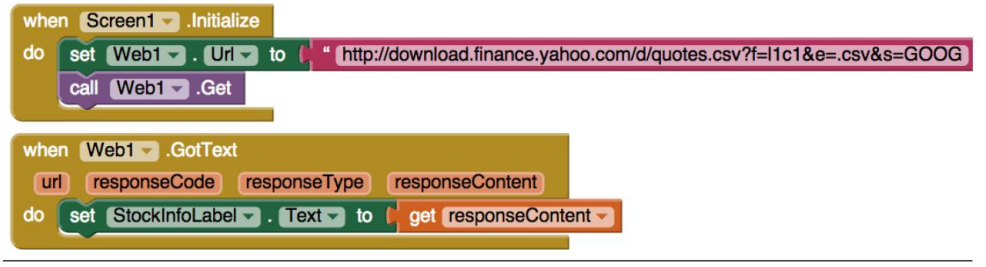
שיטות פרסום להכנסת מידע לאפליקציה שלך. המידע מגיע לא כדף הניתן לתצוגה, אלא כנתונים שאתה יכול להציג או לעבד כרצונך.
הרכיב ברמה נמוכה למדי, והשימוש בו דורש מומחיות מסוימת בתכנות. בדרך כלל אתה מגדיר את המאפיין Web.URLכדי לציין עם איזה שירות אינטרנט תתקשר, ואז אתה קורא לאחת משיטות HTTP-הכדי לבקש פעולה כלשהי. זה מסובך כי אתה צריך להבין את ה-API של שירות האינטרנט (הפרוטוקול לתקשורת), וצריך להבין איך לעבד את המידע ששירות האינטרנט מחזיר לאפליקציה שלך. עיבוד זה ידוע בשם ניתוח, וזוהי טכניקת תכנות מתקדמת.

בפרק זה, תתוודע לרכיב האינטרנט באמצעות דוגמה פשוטה יחסית הניגשת למידע על מחירי מניות פיננסיים מממשק APIציבורי שזמין על ידי .Yahoo Financeהפרוטוקול לדבר עם API-ההזה הוא פשוט למדי, והנתונים המוחזרים נמצאים ברשימת ערכים מופרדים בפסיקים (ערכים מופרדים בפסיקים, או ,CSVכך שהוא משמש כהקדמה נחמדה לתקשורת .API

לרוע המזל, לרוב ממשקי API-היש סכימות הרשאות וממשקי APIמסובכים, ולעתים קרובות הם מחזירים נתונים בפורמטים כגון JavaScript Object Notation (JSON)או ,XMLהדורשים קוד מתקדם כדי לנתח.

מדגם שוק המניות

איור 24-6מציג את החסימות של אפליקציה שמציגה מידע על מניות Googleכאשר האפליקציה מופעלת.



איור .24-7גישה למידע על מלאי חיים באמצעות רכיב האינטרנט

On Screen.Initialize, Web1.Urlמוגדר לכתובת ה-LRU לתקשורת עם Yahoo לְמַמֵן. כאשר Web1.Getנקרא, הבקשה מבוצעת, אך לא מוחזרים נתונים מיידית.

במקום זאת, כאשר יאהו מחזירה את הנתונים המבוקשים לאפליקציה שלך, אירוע Web1.GotTextמופעל, וכאן תוכל לעבד את הנתונים המוחזרים. פרמטר האירוע responseContentמחזיק את הנתונים. כאמור, ממשק APIשל Yahoo Financeמחזיר נתונים בפורמט .CSVאם תבנה את האפליקציה הזו ותפעיל אותה, תראה שהנוכחי

מחיר מניית Googleוהשינוי במחיר היום מוצגים ב- StockInfoLabel, מופרדים בפסיקים.

אתה יכול להתאים אישית את ה-lrU.beW כדי לקבל את המידע עבור חברה (או חברות) שונות וכדי לקבל סוגים שונים של מידע על שוק המניות. ה-API של Yahoo Finance, בכתובת https://code.google.com/p/yahoo-fnance-managed/wiki/CSVAPI, מפרט כיצד תוכל לשנות את כתובת האתר כדי להתאים אישית את הבקשה שלך, כמו גם את פורמט הנתונים שהיא החזרות.

ממשקי API תואמי TinyWebDB ו-BDbeWyniT

רכיב האינטרנט מספק שיטה לגישה לממשקי API. אם ממשק API פשוט למדי, כמו Finance, Yahoo מתכנתים מתחילים יכולים להשתמש ברכיב האינטרנט כדי לגשת אליו ישירות. אבל ממשקי API אחרים, כמו ה-API של Amazon שהוצג בפרק 13, הם מסובכים יותר.

עבור ממשקי API מסובכים, מתכנת מנוסה יכול להגדיר שירות אינטרנט תואם TinyWebDB שיוכל לשמש מתכנתי App Inventor פחות מנוסים כדי לגשת ל-API. לכאשר שירות כזה מוגדר, מתכנתים אחרים יכולים לגשת לשירות האינטרנט עם פרוטוקול תג-ערך הפשוט הטמון בפונקציה TinyWebDB.GetValue . אתה שולח תג מסוים כפרמטר, ואובייקט רשימה או טקסט מוחזר כערך. בדרך זו, מתכנת App Inventor מגן מהתכנות הקשה הנדרש לניתוח (להבין ולחלץ נתונים) פורמטים סטנדרטיים של נתונים כגון XML או JSON.

"תואם TinyWebDB" פירושו רק שירות אינטרנט העוקב אחר הפרוטוקול הצפוי של TinyWebDB: הוא מצפה לבקשה ספציפית, ומחזיר נתונים ש-BDbeWyniT יכולה להבין. שירות האינטרנט של Amazon API המשמש בפרק 13 הוא דוגמה לשירות אינטרנט כזה, והוא יכול לשמש כדוגמה למתכנתים שרוצים להגדיר שירות כזה (למשל, אם אתה מורה ומעוניין לספק גישה ל-API לכלשהו עבור התלמידים שלך).

בעבר, בניית ממשקי API הייתה קשה מכיוון שלא רק היית צריך להבין את התכנות ואת פרוטוקולי האינטרנט, אלא גם צריך להגדיר שרת שיארח את שירות האינטרנט שלך, ומסד נתונים כדי לאחסן את הנתונים. כעת, זה הרבה יותר קל מכיוון שאתה יכול למנף כלי מחשוב ענן כגון App Engine של גוגל ו- Elastic Compute Cloud של אמזון כדי לפרוס מיד את השירות שאתה יוצר. הפלטפורמות הללו לא רק יארחו את שירות האינטרנט שלך, אלא גם יאפשרו למאות משתמשים לגשת אליו לפני שיגבו ממך אגורה אחת. כפי שאתה יכול לדמיין, אתרים אלה הם ברכה גדולה לחדשנות.

> הפרטים של יצירת שירות אינטרנט תואם TinyWebDB הם מעבר לתחום של הספר הזה. אבל אם אתה מעוניין, בדוק את התיעוד והדוגמאות בכתובת http://appinventorapi.com/.

סיכום

רוב אתרי האינטרנט ואפליקציות סלולריות רבות אינן ישויות עצמאיות; כדי לבצע את עבודתם, הם מסתמכים על יכולת פעולה הדדית של אתרים אחרים. עם App Inventor, אתה יכול לבנות משחקים, חידונים ואפליקציות עצמאיות אחרות, אבל בקרוב תיתקל בבעיות הקשורות לגישה לאינטרנט. האם אני יכול לכתוב אפליקציה שאומרת לי מתי האוטובוס הבא יגיע לתחנה הרגילה שלי? האם אני יכול לכתוב אפליקציה ששולחת הודעות טקסט לתת-קבוצה מיוחדת של חבריי בפייסבוק? האם אני יכול לכתוב אפליקציה ששולחת ציוצים? App Inventor מספק שלושה רכיבים שיכולים לדבר עם האינטרנט: WebViewer להצגת דף אינטרנט חי; רכיב האינטרנט , לגישה למידע API-מ; ורכיב TinyWebDB לגישה לנתונים API-באינטרנט שעוצב במיוחד.

גישה API-ליכולה להיות מסובכת; אתה צריך לדעת את הפרוטוקול לבקשת מידע, ואתה צריך לעבד (לנתח) את הנתונים המורכבים המוחזרים לעתים קרובות. אבל התגמול עבור ללמוד איך לעשות זאת הוא גדול; האפליקציות שלך יכולות ליצור אינטראקציה עם העולם!