אין לשלוח הודעות טקסט בזמן נהיגה

פרק זה ילווה אותך דרך היצירה של ללא הודעות טקסט בזמן נהיגה, אפליקציית "משיבון טקסט" המגיבה אוטומטית להודעות טקסט שאתה מקבל בזמן נהיגה (או במשרד וכו'), מדברת הודעות טקסט בקול רם, ואפילו שולחת פרטי מיקום כחלק מהמערכת. תשובת טקסט אוטומטית. האפליקציה מדגימה כיצד אתה יכול לשלוט בכמה מהתכונות הנהדרות של טלפון אנדרואיד, כולל הודעות SMS, טקסט לדיבור, נתונים מתמשכים וחישת מיקום GPS.

בינואר 2010, המועצה הלאומית לבטיחות של ארצות הברית (NSC) הכריזה על תוצאות מחקר שמצא שלפחות 28% מכל תאונות הדרכים - קרוב ל-6.1 מיליון תאונות בכל שנה - נגרמות על ידי נהגים המשתמשים בטלפונים סלולריים, ולפחות 200,000 מהתאונות הללו התרחשו בזמן שנהגים שלחו הודעות טקסט.1 כתוצאה מכך, מדינות רבות אסרו על נהגים להשתמש בטלפונים סלולריים לחלוטין.

1 http://bit.ly/1qiH7aZ

איור 4-1. האפליקציה ללא הודעות טקסט בזמן נהיגה

דניאל פינגן, סטודנט באוניברסיטת סן פרנסיסקו שלוקח שיעור תכנות App Inventor, הגה רעיון לאפליקציה כדי לעזור במגפת הנהיגה וההודעות הטקסט. האפליקציה שהוא יצר, שמוצגת באיור 4-1, מגיבה אוטומטית (ודיבורית) לכל טקסט עם הודעה כגון "אני נוהג עכשיו, אצור איתך קשר בקרוב."

האפליקציה הורחבה מאוחר יותר כך שתעשה זאת דברו את הטקסטים הנכנסים בקול רם והוסיפו את מיקום ה-SPG של הנהג לטקסט התגובה האוטומטית, וזה הפך למדריך עבור האפליקציה אתר ממציאים.

כמה שבועות לאחר פרסום האפליקציה באתר App Inventor, State Farm Insurance יצרה אפליקציית אנדרואיד בשם On the Move, בעלת פונקציונליות דומה ל- While Driving. No Texting

אנחנו לא יודעים אם האפליקציה של דניאל או ההדרכה באתר App Inventor השפיעו על On the Move, אבל מעניין לשקול את האפשרות שאולי יש לאפליקציה שנוצרה בקורס תכנות מתחיל (על ידי סטודנט לכתיבה יוצרת, לא פחות!) היווה השראה לתוכנה בייצור המוני הזה, או לפחות תרם למערכת האקולוגית שהביאה אותה. זה בהחלט הדגים כיצד App Inventor הוריד את מחסום הכניסה כך שכל אחד עם רעיון טוב יוכל להפוך את הרעיון שלו במהירות ובזול לאפליקציה מוחשית ואינטראקטיבית. קלייב תומפסון מהמגזין Wired קלט את החידוש וכתב את זה:

> תוכנה, אחרי הכל, משפיעה כמעט על כל מה שאנחנו עושים. בחר כל בעיה מרכזית - התחממות כדור הארץ, שירותי בריאות או, במקרה של Finnegan, בטיחות בכבישים מהירים -ותוכנה חכמה היא חלק מהפתרון. עם זאת, רק חלק קטנטן של אנשים שוקל אי פעם ללמוד לכתוב קוד, מה שאומר שאנחנו לא מקישים על היצירתיות של חלק גדול מהחברה.2
>
> App Inventor עוסק בלחיצה על היצירתיות שתומפסון מזכיר, על פתיחת עולם יצירת התוכנה ל כל אחד.

מה תלמד

זוהי אפליקציה מורכבת יותר מאלו שבפרקים הקודמים, כך שתבנה אותה חתיכת פונקציונליות אחת בכל פעם, החל בהודעת התגובה האוטומטית. תלמד על:

2קלייב תומפסון, "קלייב תומפסון על קידוד להמונים", http://wrd.cm/1uT25O5

•רכיב הטקסט לשליחת טקסטים ועיבוד טקסטים שהתקבלו.

•טופס קלט לשליחת הודעת התגובה המותאמת אישית.

•רכיב מסד הנתונים TinyDBלשמירת ההודעה המותאמת גם לאחר סגירת האפליקציה.

•האירוע Screen.Initializeלטעינת התגובה המותאמת אישית כאשר האפליקציה השקות.

•רכיב TextToSpeechלהקראת טקסטים בקול רם.

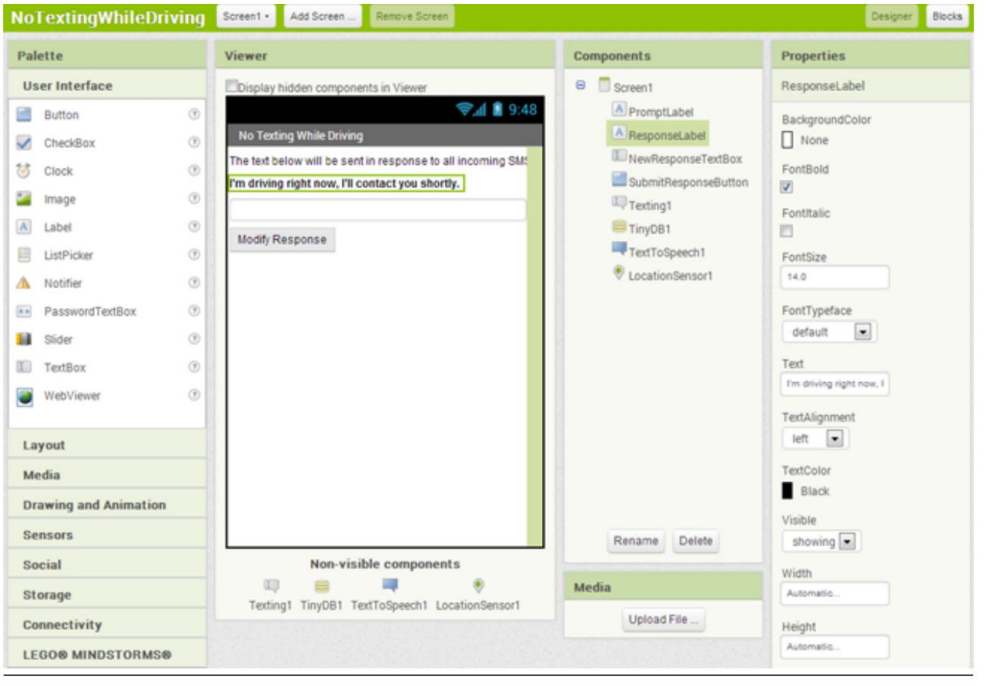
•רכיב ה- LocationSensorלדיווח על מיקומו הנוכחי של הנהג.

מתחילים

פתח את הדפדפן שלך לאתר App Inventorוהתחל פרויקט חדש. תן לזה שם "gnivirDelihWgnitxeToN" (זכור, שמות פרויקטים לא יכולים לכלול רווחים) והגדר את כותרת המסך ל"אין לשלוח הודעות טקסט בזמן נהיגה". לאחר מכן, לחץ על התחבר והגדר בדיקה חיה במכשיר שלך או באמולטור.

עיצוב הרכיבים

ממשק המשתמש של האפליקציה פשוט יחסית: יש לו תווית המורה למשתמש כיצד האפליקציה פועלת, תווית המציגה את הטקסט שאמור להישלח אוטומטית בתגובה לטקסטים נכנסים, תיבת טקסט לשינוי התגובה, ו- כפתור להגשת השינוי. תצטרך גם לגרור פנימה רכיב טקסט , רכיב , TinyDBרכיב TextToSpeechורכיב חיישן , Locationכולם יופיעו באזור "רכיבים שאינם נראים". אתה יכול לראות איך זה אמור להיראות בתמונת המצב של מעצב הרכיבים באיור .4-2



איור 4-2. האפליקציה ללא הודעות טקסט בזמן נהיגה ב-Component Designer

אתה יכול לבנות את ממשק המשתמש המוצג באיור 4-2 על ידי גרירת ה-רכיבים המפורטים בטבלה 4-1.

טבלה 4-1. כל הרכיבים לאפליקציית No Texting

מַטָרָה	קבוצת צבעים איך תקרא לזה	סוג רכיב
תן למשתמש לדעת כיצד האפליקציה פועלת.	ממשק משתמש PromptLabel	תווית
התגובה שתישלח בחזרה ל שׁוֹלֵחַ.	ממשק משתמש ResponseLabel	תווית
ממשק משתמש newResponseTextBox המשתמש יזין כאן את התגובה המותאמת אישית.		תיבת טקסט
ממשק משתמש SubmitResponseButton המשתמש לוחץ על זה כדי לשלוח תגובה.		לַחְצָן
עבדו את הטקסטים.	שליחת הודעות טקסט 1	חֲבִרְתִי	הודעות טקסט
אחסן את התגובה במסד הנתונים.	TinyDB1	אִחסוֹן	TinyDB
דבר את הטקסט בקול רם.	TextToSpeech1	מדיה	TextToSpeech
חוש היכן המכשיר נמצא.	חיישן מיקום 1	חיישני חיישן מיקום

הגדר את המאפיינים של הרכיבים בצורה הבאה:

•הגדר את הטקסט של PromptLabelל"הטקסט למטה יישלח כתגובה לכולם הודעות SMSהתקבלו בזמן שהאפליקציה הזו פועלת."

•הגדר את הטקסט של ResponseLabelל"אני נוהג עכשיו, אצור איתך קשר בְּקָרוֹב." בדוק את תכונת ההעזה שלו.

עבור הגדר את הטקסט של NewResponseTextboxל- ." "(זה משאיר את תיבת הטקסט ריקה קלט המשתמש.)

•הגדר את הרמז של NewResponseTextboxל"הזן טקסט תגובה חדש".

•הגדר את הטקסט של SubmitResponseButtonל"שנה תגובה".

הוספת התנהגויות לרכיבים

תתחיל בתכנות התנהגות התגובה האוטומטית שבה תשובת טקסט נשלחת לכל טקסט נכנס. לאחר מכן תוסיף בלוקים כדי שהמשתמש יוכל לציין תגובה מותאמת אישית ולשמור את התגובה הזו בהתמדה. לבסוף, תוסיף בלוקים שקוראים בקול את הטקסטים הנכנסים ותוסיף מידע מיקום לטקסטים של התגובה האוטומטית.

תגובה אוטומטית לטקסט

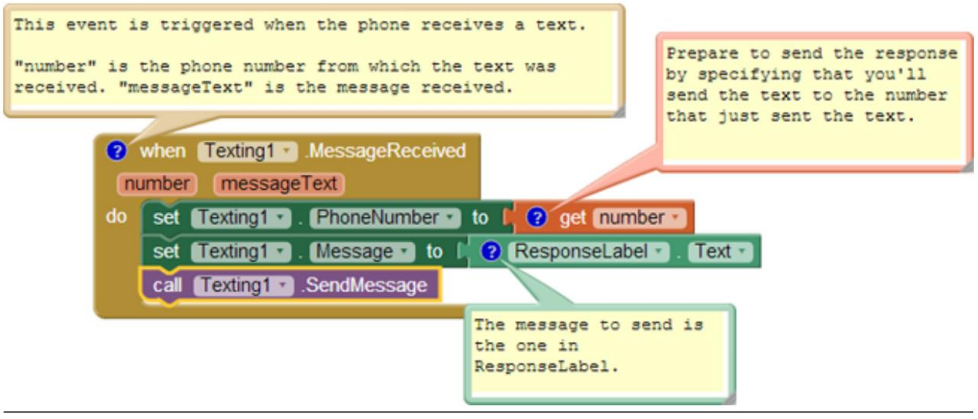
עבור התנהגות התגובה האוטומטית, תשתמש ברכיב הטקסט של .App Inventorאתה יכול לחשוב על הרכיב הזה כעל אדם קטן בתוך הטלפון שלך שיודע לקרוא ולכתוב טקסטים. לקריאת טקסטים, הרכיב מספק בלוק אירוע . Texting.MessageReceivedאתה יכול לגרור את הבלוק הזה החוצה ולהציב בתוכו בלוקים כדי להראות מה צריך לקרות כשמתקבל טקסט. במקרה של אפליקציה זו, אנו רוצים לשלוח בחזרה הודעת טקסט באופן אוטומטי בתגובה.

אתה יכול לשלוח טקסט עם שלושה בלוקים. ראשית, אתה מגדיר את מספר הטלפון שאליו יש לשלוח את הטקסט, שהוא מאפיין של רכיב . Texting1לאחר מכן, אתה מגדיר את ההודעה להישלח, גם היא מאפיין של .SMS1לבסוף, אתה למעשה שולח את הטקסט עם בלוק . Texting1.SendMessageטבלה 4-2מפרטת את כל הבלוקים שתזדקקו להתנהגות התגובה האוטומטית הזו, ואיור 4-3מראה כיצד הם צריכים להיראות בעורך הבלוקים.

טבלה .4-2החסימות לשליחת תגובה אוטומטית

מַטָרָה	מִגבָּלָה	סוג בלוק
המטפל באירועים שמופעל כאשר הטלפון מקבל הודעת טקסט.	הודעה	התקבלה הודעת טקסט .1
הגדר את המאפיין PhoneNumberלפני השליחה.		הגדר את Texting1.PhoneNumberל-SMS
גרור ממתי חסימה מספר הטלפון של האדם ששלח את הטקסט.		מספר ערך

סוג בלוק	מִגרָה	מַטָרָה
הגדר ל-1SMS.הודעה ל	הודעות טקסט	הגדר את מאפיין ההודעה לפני השליחה.
ResponseLabel.Text	ResponseLabel	ההודעה שהמשתמש הזין.
שליחת הודעות .1שלח הודעה	הודעות טקסט	שלח את ההודעה.



איור .4-3תגובה לטקסט נכנס

איך הבלוקים עובדים

כאשר הטלפון מקבל הודעת טקסט, האירוע Texting1.MessageReceivedהוא מופעל. מספר הטלפון של השולח נמצא במספר הארגומנט, וה- ההודעה שהתקבלה נמצאת בארגומנט .messageText

מכיוון שטקסט התגובה האוטומטית צריך להישלח בחזרה אל שולח, הודעות טקסט. PhoneNumberמוגדר למספר. הוגדרה הודעות טקסט ל- ,ResponseLabel.Textשזה מה שהקלדת בזמן המעצב: "אני נוהג ברגע זה, אצור איתך קשר בקרוב." כאשר אלה מוגדרים, האפליקציה מתקשרת Texting.SendMessageכדי לשלוח את התגובה בפועל.

בדוק את האפליקציה שלך תצטרך שני טלפונים כדי לבדוק התנהגות זו, אחד כדי להפעיל את האפליקציה ואחד כדי לשלוח את הטקסט הראשוני. אם לא אם יש לך טלפון שני בהישג יד, אתה יכול להשתמש Google Voice-באו א שירות דומה במחשב שלך ולשלוח ממנו הודעות טקסט שירות לטלפון שמריץ את האפליקציה. אחרי שתגדיר את הדברים, שלח הודעת טקסט לטלפון שמריץ את האפליקציה. עושה את הטלפון הראשון לקבל את טקסט התגובה?

הזנת תגובה מותאמת אישית

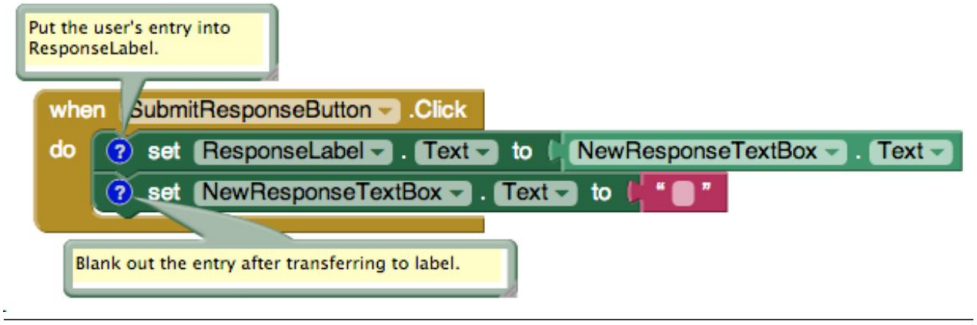
לאחר מכן, בואו נוסיף בלוקים כדי שהמשתמש יוכל להזין תגובה מותאמת אישית משלה. בתוך ה
מעצב רכיבים, הוספת רכיב TextBox בשם NewResponseTextbox;
זה המקום שבו המשתמש יקליד את התגובה המותאמת אישית. כאשר המשתמש לוחץ על
SubmitResponseButton, עליך להעתיק את הערך (NewResponseTextbox) לתוך
ResponseLabel, המשמש להגיב לטקסטים. טבלה 4-3 מפרטת את הבלוקים שתזדקק להם
להעברת תגובה שהוזנה לאחרונה ל- ResponseLabel.

טבלה 4-3. בלוקים להצגת התגובה המותאמת אישית

סוג בלוק	מִגרָה	מַטָרָה
SubmitResponseButton.Click	SubmitResponseButton.Click	המשתמש לוחץ על כפתור זה כדי לשלוח הודעה חדשה הודעת תגובה.
set ResponseLabel.Text to	set ResponseLabel.Text to	ResponseLabel העבר (הגדר) את ערך הקלט ה
NewResponseTextbox.Text	NewResponseTextbox.Text	NewResponseTextbox המשתמש הזין כאן את
הגדר את NewResponseTextbox-ל NewResponseTextbox.Text	set NewResponseTextbox.Text to	ריק את תיבת הטקסט לאחר ההעברה מֵידָע
טֶקסט	טקסט ("")	הטקסט הריק.

איך הבלוקים עובדים

תחשוב על האופן שבו אתה מקיים אינטראקציה עם טופס קלט טיפוסי: אתה קודם כל מקליד משהו בטקסט
ולאחר מכן לחץ על לחצן שלח כדי לאותת למערכת לעבד אותו. טופס הקלט
שכן האפליקציה הזו אינה שונה. איור 4-4 מראה כיצד הבלוקים מתוכננים כך
כאשר המשתמש לוחץ על SubmitResponseButton.Click , SubmitResponseButton
האירוע מופעל.



איור 4-4. הגדרת התגובה לכניסת המשתמש

המטפל באירועים במקרה זה מעתיק (או, במונחי תכנות, מגדיר) את מה שהמשתמש הזין ב-NewResponseTextbox ל- ResponseLabel. זכור ש- ResponseLabel מחזיקה את ההודעה שתישלח בתגובה האוטומטית, כך שברצונך להיות בטוח שתמקם שם את ההודעה המותאמת החדשה שהוזנה.

בדוק את האפליקציה שלך הזן תגובה מותאמת אישית ושלח אותה, ולאחר מכן השתמש בטלפון השני כדי לשלוח טקסט נוסף לטלפון שמריץ את האפליקציה. האם התגובה המותאמת אישית נשלחה?

אחסון התגובה המותאמת אישית באופן עקבי

המשתמש שלך יכול כעת להתאים אישית את התגובה האוטומטית, אבל יש מלכוד אחד: אם המשתמש יזין תגובה מותאמת אישית ולאחר מכן סוגר את האפליקציה ומפעיל אותה מחדש, התגובה המותאמת אישית לא תופיע (במקום זאת, תגובת ברירת המחדל תופיע). התנהגות זו אינה מה שהמשתמשים שלך מצפים; הם ירצו לראות את התגובה המותאמת אישית שהזינו כשהם יפעילו מחדש את האפליקציה. כדי לגרום לזה לקרות, עליך לאחסן את התגובה המותאמת אישית הזו בהתמדה.

הצבת נתונים במאפיין ResponseLabel.Text מאחסנת אותם מבחינה טכנית, אך הבעיה היא שהנתונים המאוחסנים במאפייני הרכיב הם נתונים חולפים. נתונים חולפים הם כמו הזיכרון לטווח קצר שלך; הטלפון "שוכח" אותו ברגע שאפליקציה נסגרת. אם אתה רוצה שהאפליקציה שלך תזכור משהו בהתמדה, אתה צריך להעביר אותו מזיכרון לטווח קצר (מאפיין רכיב או משתנה) לזיכרון לטווח ארוך (מסד נתונים או פלי).

כדי לאחסן נתונים באופן מתמשך, App Inventor-באתה משתמש ברכיב TinyDB, המאחסן נתונים בפליסה במכשיר האנדרואיד. TinyDB מספקת שתי פונקציות: StoreValue ו- GetValue. עם הראשון, האפליקציה יכולה לאחסן מידע במסד הנתונים של המכשיר, ואילו עם השנייה, האפליקציה יכולה לאחזר מידע שכבר נשמר.
עבור אפליקציות רבות, תשתמש בסכימה הבאה:

1. אחסן נתונים במסד הנתונים בכל פעם שהמשתמש שולח ערך חדש.
2. כאשר האפליקציה מופעלת, טען את הנתונים ממסד הנתונים למשתנה או תכונה.

תתחיל בשינוי המטפל באירועים SubmitResponseButton.Click כך שהוא מאחסן את הנתונים באופן מתמיד, באמצעות הבלוקים המפורטים בטבלה 4-4.

טבלה 4-4. בלוקים לאחסון התגובה המותאמת אישית עם TinyDB

מַטָרָה	שם בלוק	תמונה
אחסן את ההודעה המותאמת אישית במסד הנתונים של הטלפון.	TinyDB1	TinyDB1.StoreValue

סוג בלוק	מִגרָה	מַטָרָה
ResponseLabel טקסט("responseMessage")		טקסט ResponseLabel.Text הודעת התגובה כאן.

איך הבלוקים עובדים

האפליקציה הזו משתמשת ב- TinyDB כדי לקחת את הטקסט שזה עתה שמה ב- ResponseLabel ולאחסן אותו במסד הנתונים. כפי שמוצג באיור 4-5, כאשר אתה מאחסן משהו במסד הנתונים, אתה מספק איתו תג; במקרה זה, התג הוא "responseMessage". חשבו על התג כשם לנתונים במסד הנתונים; זה מזהה באופן ייחודי את הנתונים שאתה מאחסן. כפי שתראה בסעיף הבא, תשתמש באותו תג ("responseMessage") כאשר אתה טוען את הנתונים חזרה ממסד הנתונים.

איור 4-5. אחסון התגובה המותאמת אישית באופן מתמיד

אחזור התגובה המותאמת אישית כאשר האפליקציה נפתחת

הסיבה לאחסון התגובה המותאמת במסד הנתונים היא כדי שניתן יהיה לטעון אותה בחזרה לאפליקציה בפעם הבאה שהמשתמש יפתח אותה. App Inventor מספק בלוק אירועים מיוחד שמופעל כאשר האפליקציה נפתחת: מסך .1אתחול (אם השלמת את MoleMash בפרק 3, ראית את זה בעבר). אם תגרור את חסימת האירוע החוצה ותציב בו בלוקים, החסימות הללו יבוצעו מיד עם הפעלת האפליקציה.

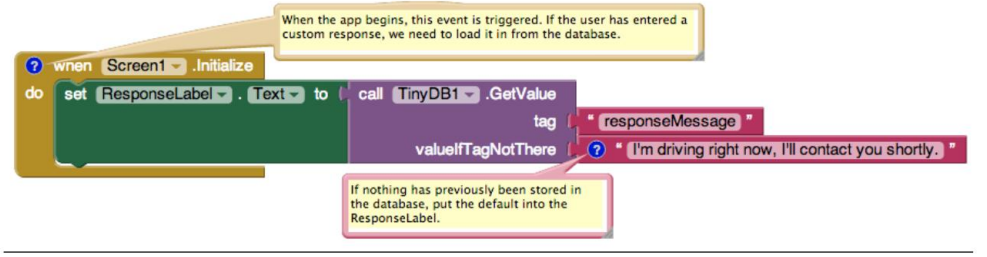
עבור אפליקציה זו, המטפל באירוע Screen1.Initialize שלך יטען את התגובה המותאמת אישית ממסד הנתונים באמצעות הפונקציה TinyDB.GetValue . הבלוקים שתזדקקו לכך מוצגים בטבלה 4-5.

טבלה 4-5. חסימות לטעינת הנתונים בחזרה בעת פתיחת האפליקציה

מַטָרָה	מְגִרָה	סוג בלוק
זה מופעל כאשר האפליקציה מתחילה.	מסך 1	מסך 1. אתחול
קבל את טקסט התגובה המאוחסן ממסד הנתונים.	TinyDB1	TinyDB1.GetValue
חבר את זה לשקע התג של TinyDB.GetValue, מה שהופך בטוח שהטקסט זהה לזה שבו נעשה שימוש ב TinyDB.StoreValue מוקדם יותר.	טֶקסט	טקסט ("responseMessage")
חבר את זה לחריץ valueIfTagNotThere של TinyDB.GetValue. זו הודעת ברירת המחדל שצריך לשמש אם המשתמש עדיין לא אחסן תגובה מותאמת אישית.	טֶקסט	טקסט ("אני נוהג עכשיו, אני אעשה צור איתך קשר בהקדם")
הצב set ResponseLabel.Text ל-ResponseLabel את הערך שאוחזר ב-ResponseLabel.		

איך הבלוקים עובדים

איור 4-6 מציג את הבלוקים. כדי להבין אותם, עליך לדמיין פתיחת משתמש האפליקציה בפעם הראשונה, הזנת תגובה מותאמת אישית ופתיחת האפליקציה פעמים שלאחר מכן. בפעם הראשונה שהמשתמש פותח את האפליקציה, לא יהיה שום התאמה אישית תגובה במסד הנתונים כדי לטעון, אז אתה רוצה להשאיר את תגובת ברירת המחדל ב- ResponseLabel. בהשקות עוקבות, אתה רוצה לטעון את המאוחסן קודם לכן תגובה מותאמת אישית ממסד הנתונים והצב אותה ב- ResponseLabel.



איור 4-6. טעינת התגובה המותאמת אישית ממסד הנתונים עם אתחול האפליקציה

כאשר האפליקציה מתחילה, האירוע Screen1.Initialize מופעל. האפליקציה קוראת ל- TinyDB1.GetValue עם תג של responseMessage, אותו תג שבו השתמשת כאשר אתה אחסן את ערך התגובה המותאמת אישית של המשתמש קודם לכן. אם יש נתונים ב- TinyDB עם תג של responseMessage, הוא מוחזר וממוקם ב- ResponseLabel.
עם זאת, לא יהיו נתונים בפעם הראשונה שבה האפליקציה תופעל; זה יהיה ה רישיות עד שהמשתמש מקליד תגובה מותאמת אישית. כדי לטפל במקרים כאלה, TinyDB1.GetValue יש פרמטר שני, valueIfTagNotThere. אם לא נמצא נתונים, הערך ב valueIfTagNotThere משמש במקום זאת. במקרה הזה, "אני נוהג עכשיו, אצור איתך קשר בקרוב," ערך ברירת המחדל, ממוקם ב- ResponseLabel.

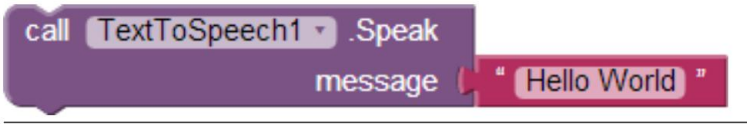
בדוק את האפליקציה שלך כדי לבדוק התנהגות זו, עליך להפעיל מחדש את האפליקציה שלך כדי לראות אם הנתונים באמת מאוחסנים באופן מתמשך ומאוחזרים כהלכה. בבדיקה חיה, אתה יכול להפעיל מחדש את האפליקציה על ידי שינוי מאפיין רכיב כלשהו במעצב, כגון גודל הגופן של תווית. זה יגרום לטעינה מחדש של האפליקציה ולהפעלת .Screen.Initializeכמובן, אתה יכול גם לבדוק את האפליקציה על ידי בנייתה בפועל והתקנת .apk fle-ה בטלפון שלך. לאחר שהאפליקציה נמצאת בטלפון שלך, הפעל אותה, הקלד הודעה עבור התגובה המותאמת אישית, סגור את האפליקציה ולאחר מכן פתח אותה מחדש. אם ההודעה שהזנת עדיין שם, הדברים פועלים כהלכה.

דיבור הטקסטים הנכנסים בקול רם

בחלק זה, תשנה את האפליקציה כך שכאשר תקבל הודעת טקסט, מספר הטלפון של השולח, יחד עם ההודעה, יושמע בקול. הרעיון כאן הוא שכאשר אתה נוהג ושומע טקסט נכנס, אתה עלול להתפתות לבדוק את הטקסט גם אם אתה יודע שהאפליקציה שולחת תגובה אוטומטית. עם טקסט לדיבור, אתה יכול לשמוע את הטקסטים הנכנסים ולהחזיק את הידיים על ההגה.

מכשירי אנדרואיד מספקים יכולות טקסט לדיבור, App Inventor-ומספק רכיב, TextToSpeech, שידבר כל טקסט שתיתן לו. שימו לב שה"טקסט" ב- TextToSpeechמתייחס לרצף של אותיות, ספרות וסימני פיסוק, לא לטקסט .SMS.

רכיב TextToSpeechהוא פשוט מאוד לשימוש. אתה פשוט קורא לפונקציית Speakשלה וחבר את הטקסט שברצונך להשמיע לתוך חריץ ההודעות שלו. לדוגמה, הבלוקים המוצגים באיור 7-4יגידו את המילים "Hello World."



איור 7-4בלוקים לדבר "שלום עולם" בקול.

עבור האפליקציה ללא הודעות טקסט בזמן נהיגה, תצטרך לספק אפליקציה מסובכת יותר הודעה לדיבור, כזו הכוללת גם את הטקסט שהתקבל וגם את מספר הטלפון של האדם ששלח אותה. במקום לחבר אובייקט טקסט סטטי כמו בלוק הטקסט "Hello World", תחבר בלוק הצטרפות . פונקציה נפוצה, join, מאפשרת לך לשלב פיסות טקסט נפרדות (או מספרים ותווים אחרים) לאובייקט טקסט אחד.

תצטרך לבצע את השיחה ל- TextToSpeech.Speakבתוך Texting.MessageReceivedמטפל באירועים שתכנת קודם לכן. הבלוקים שתכנתת בעבר מטפלים באירוע זה על ידי הגדרת מספר הטלפון וההודעה

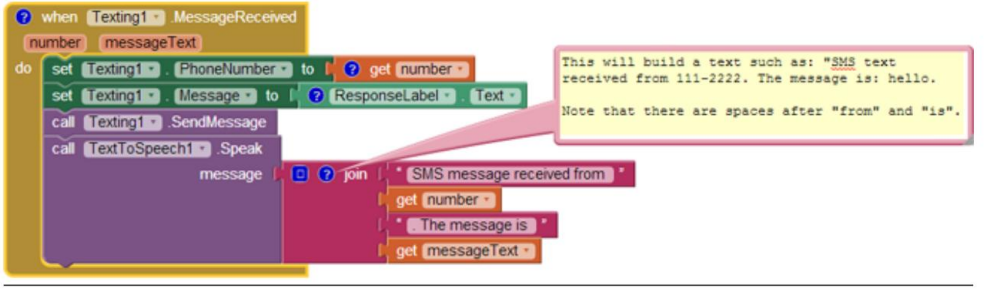
מאפיינים של רכיב הטקסטים בצורה מתאימה ולאחר מכן שליחת התגובה
טֶקסט. אתה תרחיב את המטפל באירועים על ידי הוספת הבלוקים המפורטים בטבלה 4-6.

טבלה 4-6. חסימות לקריאת הטקסט הנכנס בקול

סוג בלוק	מַטָרָה	מִגֵרָה	
TextToSpeech1.Speak	אמור (דבר יחד) את המילים שאמרו.		
Texting1.SendMessage	שרשר (חבר יחד) את המילים שאמרו.		
set Texting1.Message to	קבע את (הטקסט) של (Texting1) ל- (טקסט)		
get messageText	קבל את (הטקסט) של (messageText)		
when Texting1.MessageReceived number messageText	שים פקודה אחרי מספר הטלפון ואז אמור, "ה טקסט ("ההודעה היא") ההודעה היא."		
קבל הודעהטקסט	גרור פנימה מרגע החסימה ההודעה המקורית התקבלה.		

איך הבלוקים עובדים

לאחר שליחת התגובה, נקראת הפונקציה TextToSpeech1.Speak , כפי שמוצג ב-
התחתון של איור 4-8. אתה יכול לחבר כל טקסט לשקע ההודעות של
פונקציית TextToSpeech1.Speak . במקרה זה, הצטרף משמש לבניית המילים להיות
מדוברת -הוא משרשר (או מצטרף) יחד את הטקסט "טקסט SMS שהתקבל מ" ואת ה
מספר טלפון ממנו התקבלה ההודעה (קבל מספר), בתוספת הטקסט The."
ההודעה היא", ולבסוף ההודעה שהתקבלה (קבל messageText). אז אם הטקסט "שלום"
נשלח מהמספר "111-2222", הטלפון היה אומר, "טקסט SMS התקבל מ
111-2222. ההודעה היא שלום."



איור 4-8. אמירת הטקסט הנכנס בקול רם

בדוק את האפליקציה שלך תצטרך טלפון שני כדי לבדוק את האפליקציה שלך.
מהטלפון השני, שלח הודעת טקסט לטלפון שמריץ את
אפליקציה. האם הטלפון שמריץ את האפליקציה מדבר את הטקסט בקול?
האם זה עדיין שולח תגובה אוטומטית?

הוספת פרטי מיקום לתגובה

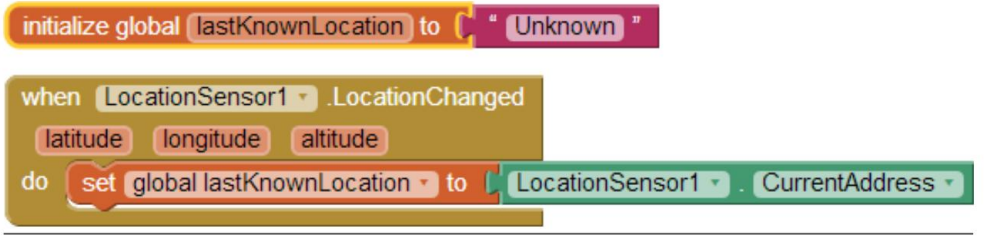
אפליקציות צ'ק-אין עוזרות לאנשים לעקוב אחר מיקומו של זה. יש פרטיות גדולה
חששות עם אפליקציות כאלה, אחת הסיבות היא שמעקב אחר מיקום מדליק את האנשים
חשש ממנגנון "האח הגדול" שממשלה טוטליטרית עלולה להקים כדי לעקוב אחריו
מקום הימצאו של אזרחיה. אבל אפליקציות שמשתמשות במידע על מיקום יכולות להיות שימושיות למדי.
תחשוב על ילד אבוד, או מטיילים שעברו על השביל ביער.
באפליקציית ללא הודעות טקסט בזמן נהיגה, אתה יכול להשתמש במעקב אחר מיקום כדי להעביר קצת
מידע נוסף בתגובה האוטומטית להודעות טקסט נכנסות. במקום רק "אני נוהג",
הודעת התגובה יכולה להיות משהו כמו, "אני נוהג ואני כרגע ב-3143
שדרת הדובדבנים." עבור מישהו שמחכה לבואו של חבר או בן משפחה, זה
מידע נוסף יכול להועיל.

App Inventorמספק את רכיב ה- LocationSensorלהתממשקות עם
ה-SPG של הטלפון (או מערכת המיקום הגלובלית). מלבד קו רוחב ואורך
מידע, חיישן המיקום יכול גם להתחבר למפות Googleכדי לספק את זה של הנהג
כתובת הרחוב הנוכחית.

חשוב לציין של- LocationSensorלא תמיד יש קריאה. לזה
סיבה, אתה צריך לדאוג להשתמש ברכיב כראוי. ליתר דיוק, האפליקציה שלך
צריך להגיב למטפל באירועים . LocationSensor.LocationChangedא
אירוע LocationChangedמתרחש כאשר חיישן המיקום של הטלפון מקבל תחילה קריאה,
וכאשר הטלפון מועבר ליצירת קריאה חדשה. שימוש בלוקים המפורטים ב
טבלה ,4-7התכנית שלנו, המוצגת באיור ,4-9תגיב לאירוע LocationChanged
על ידי הצבת הכתובת הנוכחית במשתנה, נציין את שם .lastKnownLocationמאוחר יותר, נעשה
שנה את הודעת התגובה כך שתכלול את הכתובת שאנו מקבלים מהמשתנה הזה.

טבלה .4-7בלוקים להגדרת חיישן המיקום

מַטָרָה	מִגֵרָה	סוג בלוק
צור משתנה שיחזיק את הקריאה האחרונה כתובת.	משתנים	אתחול משתנה גלובלי ("מיקום אחרון ידוע")
הגדר את ערך ברירת המחדל למקרה של הטלפון החיישן לא עובד.	טֶקסט	טקסט ("לא ידוע")
זה מופעל במיקום הראשון קריאה וכל שינוי מיקום.	חיישן מיקום 1	LocationSensor1.LocationChanged
הגדר את המשתנה הזה לשימוש מאוחר יותר.	גרור מאתחול בלוק גלובלי.	הגדר את lastKnownLocationהגלובלי ל
זוהי כתובת רחוב כגון 2222" רחוב ווילארד, אטלנטה, ג'ורג'יה."	חיישן מיקום 1	LocationSensor1.CurrentAddress



איור 4-9. רישום מיקום הטלפון במשתנה בכל פעם שמיקום ה-SPG נמצא חש

איך הבלוקים עובדים

האירוע LocationSensor1.LocationChanged מופעל בפעם הראשונה שהחיישן מקבל קריאת מיקום ואז בכל פעם המכשיר מועבר כך שקריאה חדשה מופק. הפונקציה LocationSensor1.CurrentAddress נקראת כדי לקבל את כתובת הרחוב הנוכחית של המכשיר ואחסן אותה במשתנה lastKnownLocation .

שימו לב שעם הבלוקים האלה סיימתם רק חצי מהעבודה. האפליקציה עדיין צריכה כדי לשלב את פרטי המיקום בטקסט התגובה האוטומטית שיישלח בחזרה לשולח. אתה תעשה את זה בשלב הבא.

שליחת המיקום כחלק מהתגובה

באמצעות המשתנה lastKnownLocation, אתה יכול לשנות את ה- Texting1.MessageReceived מטפל באירועים כדי להוסיף מידע מיקום לתגובה. טבלה 4-8 מפרטת את הבלוקים תצטרך בשביל זה.

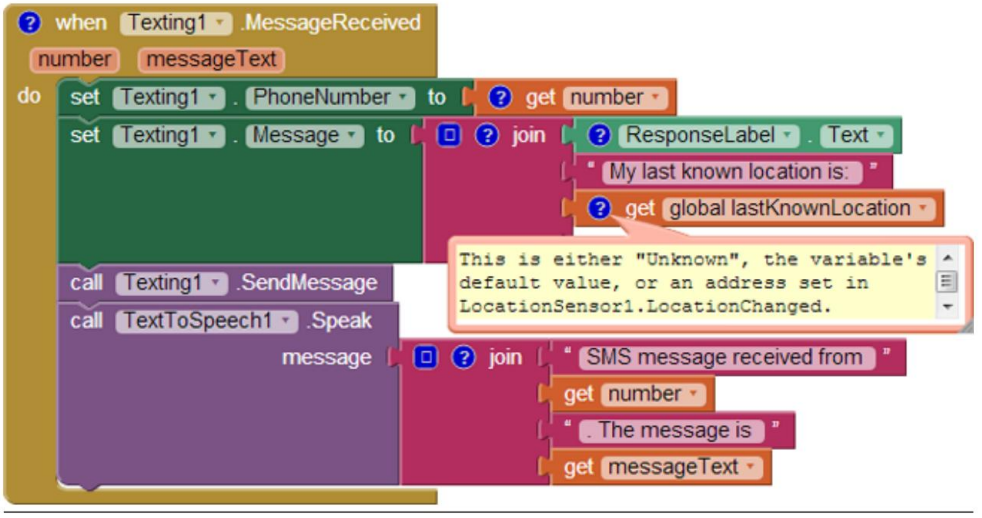
טבלה 4-8. חסימות להצגת מידע מיקום בתגובה האוטומטית

מַטָרָה	מִגרָה	סוג בלוק
לשרשר קצת טקסט ביחד	טקסט	להצטרף
זוהי ההודעה (המותאמת אישית) בתיבת הטקסט MessageTextBox.		ResponseLabel.Text
זה ייאמר לאחר ההודעה המותאמת אישית (שים לב ל חלל מוביל).	טֶקסט	טקסט ("המיקום האחרון הידוע שלי הוא:")
זוהי כתובת כגון "1600 Pennsylvania Ave NW, וושינגטון די.סי. "20500.	קבל חיישן מיקום אחרון גלובלי	

איך הבלוקים עובדים

התנהגות זו פועלת בהתאם לאירוע LocationSensor1.LocationChanged ו המשתנה lastKnownLocation. כפי שניתן לראות באיור 4-10, במקום ישירות שליחת הודעה המכילה את הטקסט ב- ResponseLabel.Text, האפליקציה תחילה בונה א

הודעה באמצעות הצטרפות. הוא משלב את טקסט התגובה ב- ResponseLabel.Textעם הטקסט "המיקום האחרון הידוע שלי הוא:" ואחריו המשתנה lastKnownLocation.



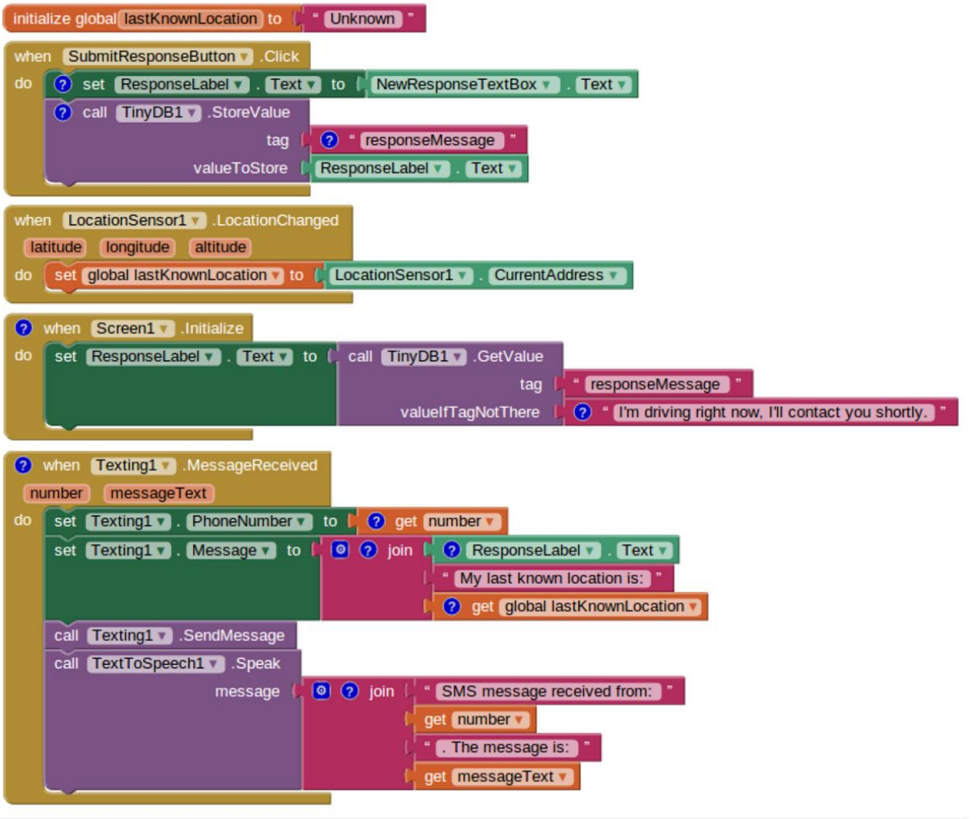
איור 4-10. כולל פרטי מיקום בטקסט התגובה

ערך ברירת המחדל של lastKnownLocationהוא "לא ידוע", אז אם חיישן המיקום עדיין לא יצר קריאה, החלק השני של הודעת התגובה יכיל את הטקסט "המיקום האחרון הידוע שלי הוא: לא ידוע." אם הייתה קריאה, החלק השני של התגובה יהיה משהו כמו, "המיקום האחרון הידוע שלי הוא: 1600 Pennsylvania Ave NW, Washington, DC 20500."

בדוק את האפליקציה שלך מהטלפון השני, שלח הודעת טקסט לטלפון שמריץ את האפליקציה. האם הטלפון השני מקבל את טקסט התגובה עם פרטי המיקום? אם לא, ודא שהפעלת את ה-SPG בהגדרות המיקום של הטלפון שמריץ את האפליקציה.

האפליקציה השלמה: אין לשלוח הודעות טקסט בזמן נהיגה

איור 4-11מציג את תצורת הבלוק הסופי עבור ללא הודעות טקסט בזמן נהיגה.



איור 4-11. האפליקציה המלאה ללא הודעות טקסט בזמן נהיגה

וריאציות

לאחר שתפעיל את האפליקציה, אולי תרצה לחקור כמה וריאציות, כגון הבאות:

•כתוב גרסה המאפשרת למשתמש להגדיר תגובות מותאמות אישית לפרט מספרי טלפון נכנסים. תצטרך להוסיף בלוקים מותנים (אם) שבודקים את המספרים האלה. למידע נוסף על בלוקים מותנים, ראה פרק 18.

•כתוב גרסה ששולחת תגובות מותאמות אישית על סמך האם המשתמש כן בתוך גבולות קווי רוחב/קו אורך מסוימים. לכן, אם האפליקציה תקבע שאתה בחדר 222, היא תשלח בחזרה "בוב נמצא בחדר 222" ואינו יכול לשלוח הודעות טקסט עכשיו." למידע נוסף על חיישן המיקום וקביעת גבולות, ראה פרק 23.

•כתוב גרסה שמשמיעה אזעקה כאשר מתקבל טקסט ממספר ב
רשימת "להודיע". לעזרה בעבודה עם רשימות, ראה פרק 19.

סיכום

להלן כמה מהמושגים שכיסינו במדריך זה:

•אתה יכול להשתמש ברכיב ה- SMSגם כדי לשלוח הודעות טקסט וגם לעבד את אלה שמתקבלות. לפני שתתקשר ל- ,Texting.SendMessageעליך להגדיר את המאפיינים PhoneNumber- ו Messageשל רכיב ה- . SMSכדי להגיב לטקסט נכנס, תכנת את המטפל . Texting.MessageReceived

•רכיב ה- TinyDBמשמש לאחסון מידע מתמשך -ב-
מסד הנתונים של הטלפון -כך שניתן לטעון מחדש את הנתונים בכל פעם שהאפליקציה נפתחת.
למידע נוסף על ,TinyDBראה פרק .22

•רכיב TextToSpeechלוקח כל אובייקט טקסט ומדבר אותו בקול רם.

•ניתן להשתמש ב- joinכדי לחבר (או לשרשר) פריטי טקסט נפרדים לאובייקט טקסט בודד.

•הרכיב LocationSensorיכול לדווח על קו הרוחב, קו האורך וכתובת הרחוב הנוכחית של הטלפון. כדי להבטיח שיש לו קריאה, עליך לגשת לנתונים שלו בתוך המטפל באירועים , LocationSensor.LocationChangedהמופעל בפעם הראשונה שמתבצעת קריאה ובכל שינוי לאחר מכן. למידע נוסף על חיישן המיקום, ראה פרק .23

אם אתה מעוניין לחקור אפליקציות לעיבוד SMSעוד יותר, בדוק את אפליקציית Broadcast Hubבפרק .11