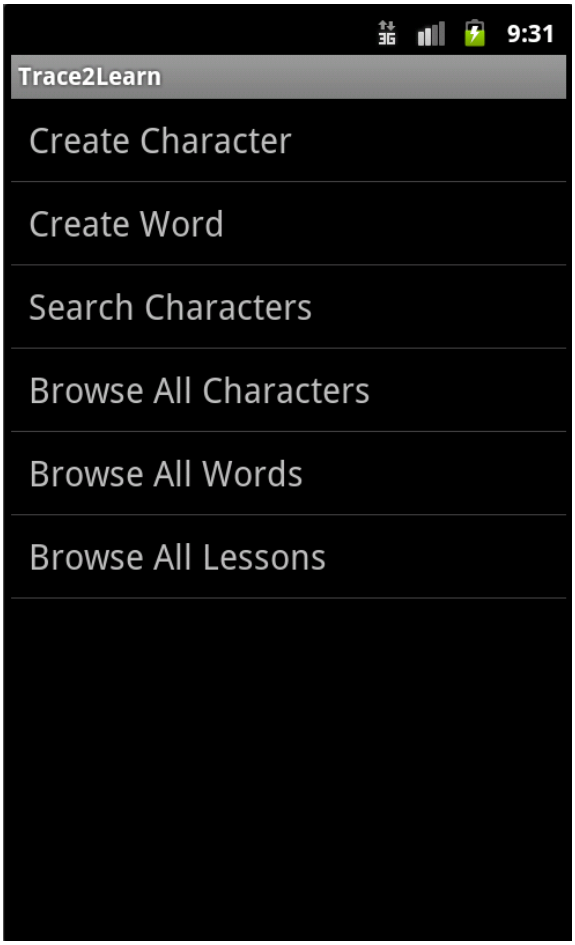
Trace2Learn Customer Handoff Document

Sam Appelbaum, Bryan Chiang, Isabel Fan, Ryan Gormley

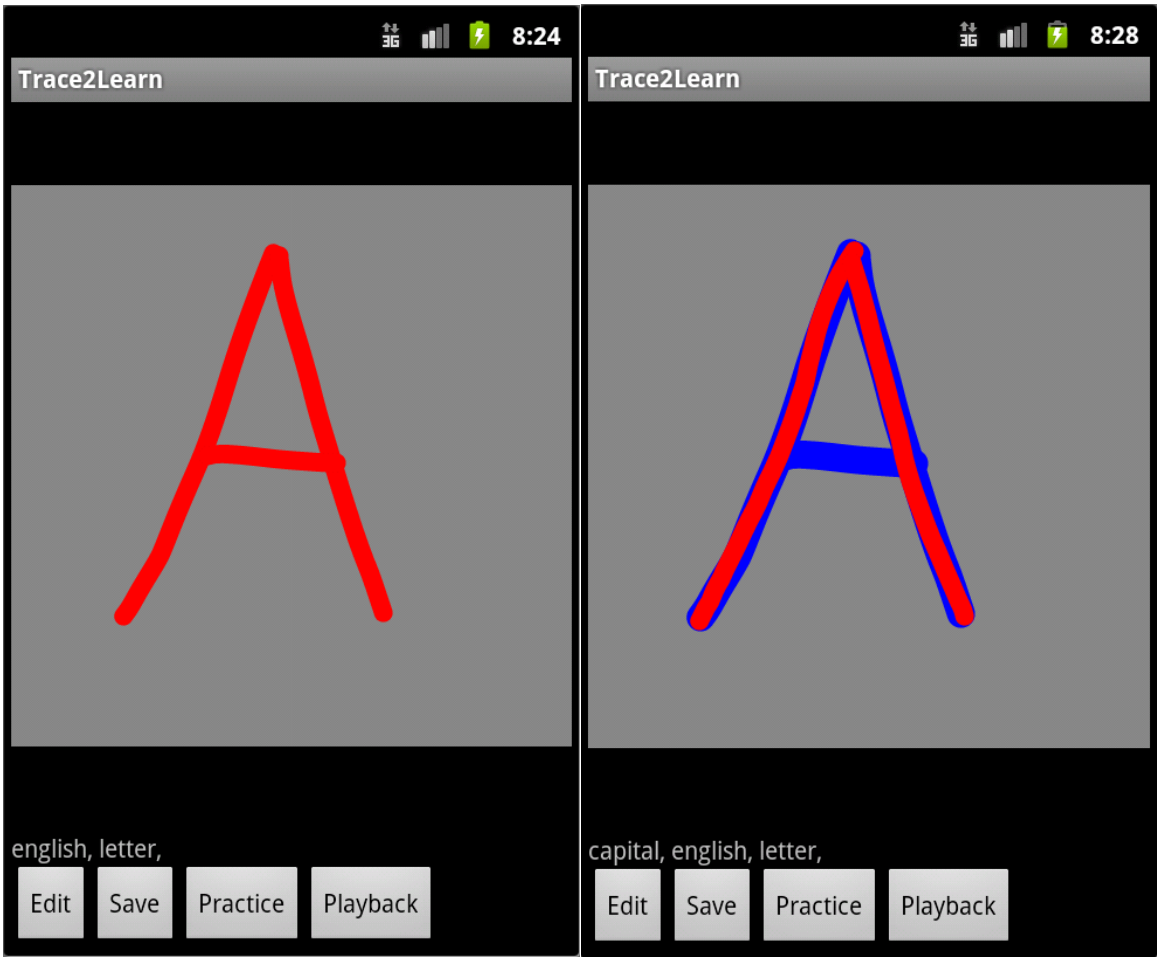
User Manual

Home Screen



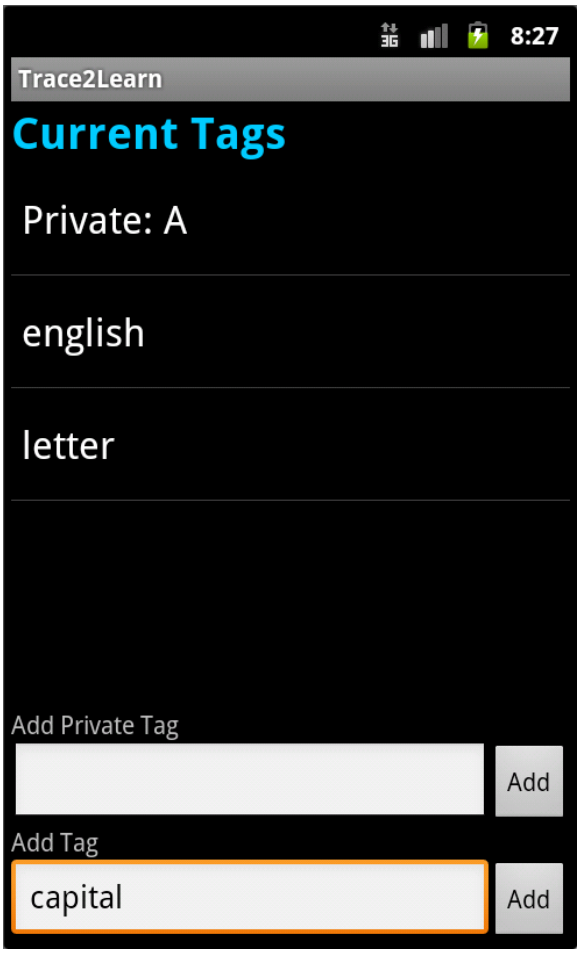
SCREENSHOT 1:

Create Character



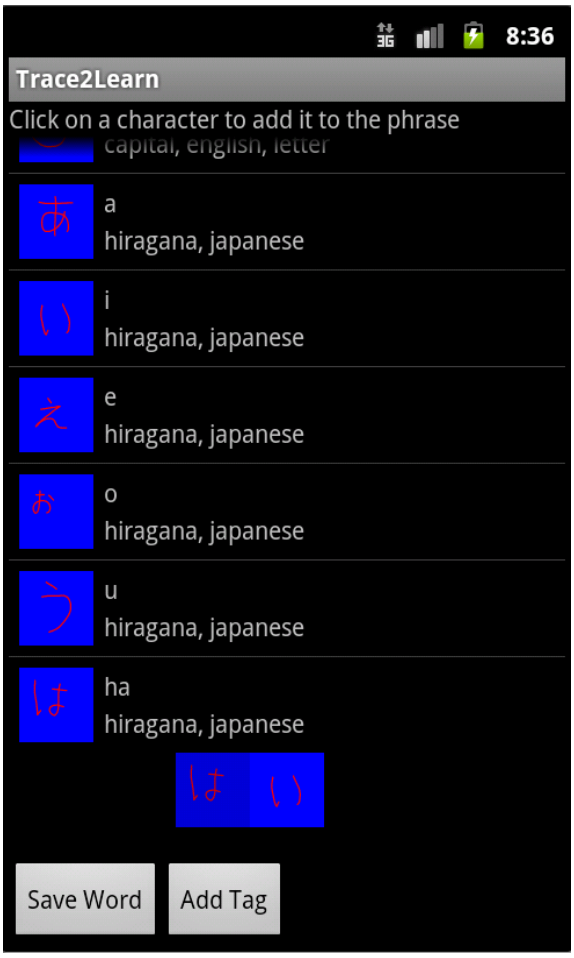
SCREENSHOT 2 (left) & 3 (right): This screen is where the user can create and practice a character. When the page loads, the activity is in edit mode, and the user can draw strokes onto the screen. Once the user is done adding strokes, they can press the *Save* button which will take the user to the tagging screen. The user can also see the character played back by pressing the playback button which will play the character stroke by stroke. THe user can also practice tracing the character, in which case the character will be drawn stroke by stroke in blue. When the user traces a stroke, the next stroke will be drawn. At any time, the user can go back and edit the strokes by pressing the edit button, after adding new strokes, the user needs to press the save button again to commit the strokes.

Tagging Characters/Words



SCREENSHOT 4: This screen allows the user to add tags to a character or word. The existing tags are displayed in the list. Each character/word can have one private tag. Adding another private tag will overwrite the current private tag. Each character/word can have multiple public tags. When a new tag is added, it will show up in the list.

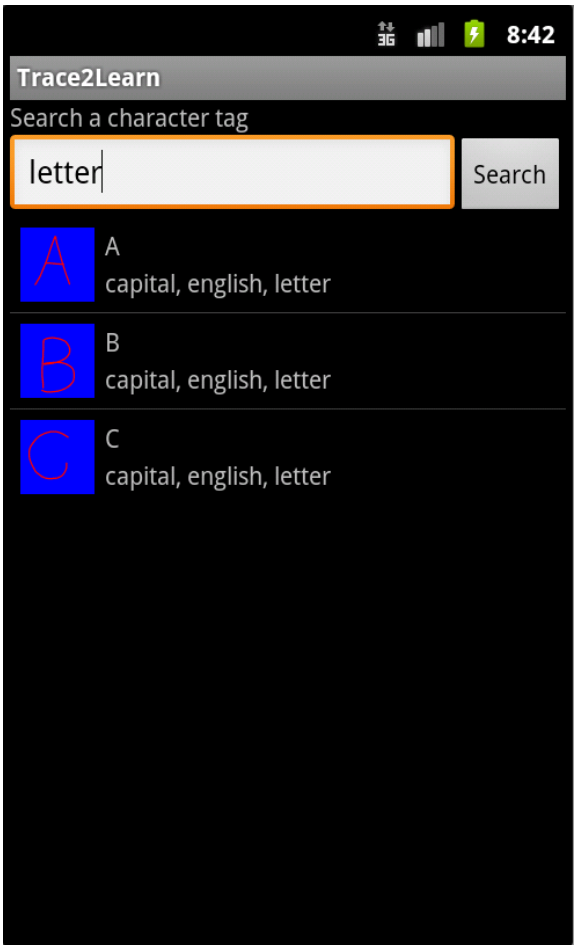
Create Word



SCREENSHOT 5: The 'Create Word' screen lists all the characters that the user has created. To add a character to the new word, the user needs to select a character from the list. Once a character is selected, it will show up at the bottom of the screen. When the user selects another character, it will be appended to the current list of characters at the bottom. When the user is done selecting characters, he can press the 'Save Word' button to save the word to the database. If the user tries to press 'Add Tag' before the word is saved, a message will popup for the user to first save the word. When the word is saved, a window will pop up asking the user to add the word to a lesson (shown below). If lessons currently exist, they will be displayed in the list. The user can select an existing lesson, or type a new one and press 'Create New' to create that lesson. If the user doesn't want to add the word to a lesson, he can press the skip button.

SCREENSHOT 6

Search Characters

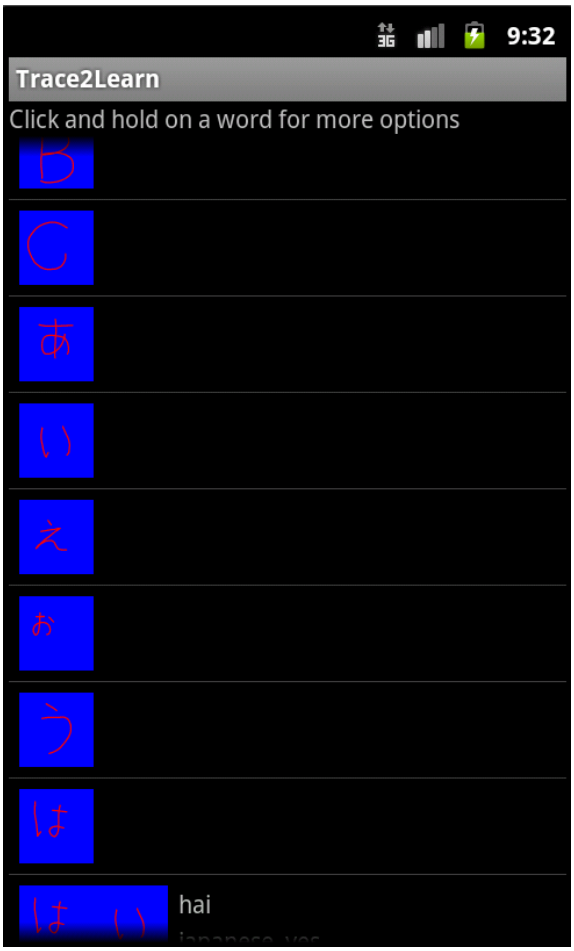


SCREENSHOT 7: This screen allows the user to search characters by public tag. The user can type a tag to search, and matching characters will show up in a list. The user can select a character to go to the creation/playback screen for that character.

Browse All Characters

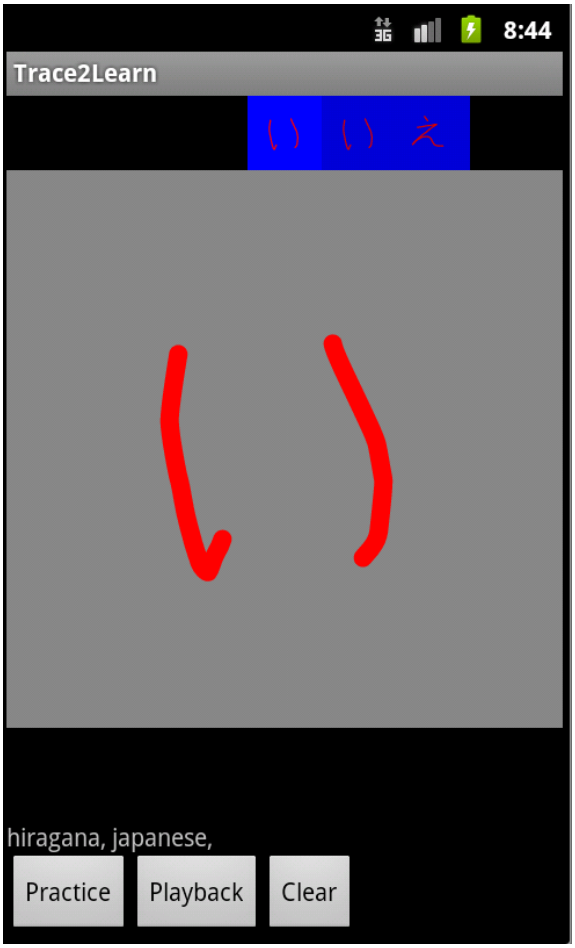
SCREENSHOT 8: This screen lets the user see all the characters that were created. The image of the character along with the private and public tags are listed in each row. Clicking on a character will bring the user to the creation/playback screen for that character. Long pressing the character will allow the user to add tags to the character or delete the character. The character will only be deleted if it is not being used by any words. If it is being used by a word, an error will pop up.

Browse All Words



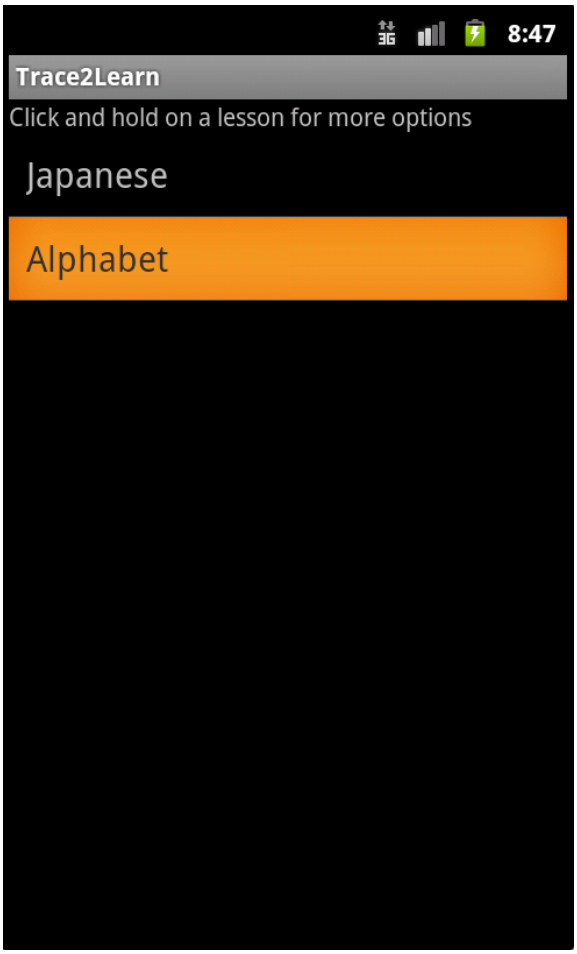
SCREENSHOT 9:This screen lets the user see all the words that were created. The image of the word is displayed, along with private and public tags. Clicking on a word will enter word practice mode, discussed below. Long pressing on the word will allow the user to add the word to a lesson, add tags to the word, or delete the word. Adding to a lesson is the same as discussed in the 'Create Word' section, with a popup that allows the user to add the word to a new or existing lesson. The word will always be deleted, even if it exists in a lesson. It will be removed from all lessons that contain that word. The individual characters will not be deleted.

Word Practice Mode



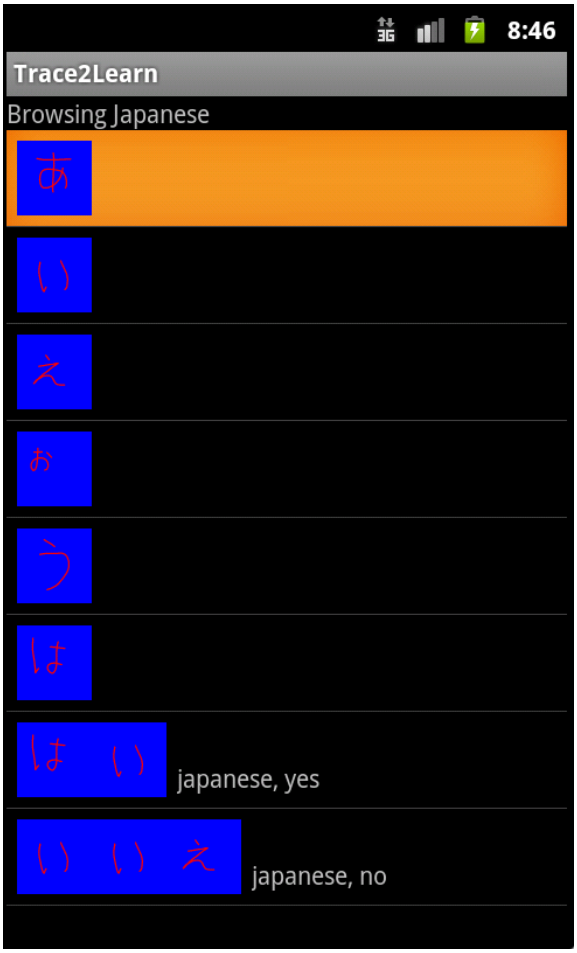
SCREENSHOT 10: This is the screen that the user can user to practice drawing strokes. Characters are displayed one by one. The user can switch between characters by pressing any of the characters shown at the top of the screen. The user can practice a character by selecting the character from the top, and then pressing the practice button, and the user can trace the character in the same manner as in the character view pane. The user can see the character drawn stroke by stroke by pressing the playback button.

Browse All Lessons



SCREENSHOT 11: This screen lets the user see all the lessons that have been created. Long pressing a lesson will allow the user to delete the lesson. Deleting a lesson only deletes the lesson, not the words associated with it. Clicking on a lesson will let the user view the words in that lesson.

Browse a Particular Lesson



SCREENSHOT 12: This screen allows the user to view the words in a particular lesson that is selected. It provides the same functionality as the 'Browse all Words' screen, but only shows the words in the particular lesson selected. The user can select a word to enter Word Practice Mode, or long press to add to a lesson, add a tag, or delete the word.

Technical Documentation

SCREENSHOT 1

Activity Class: MainMenuActivity

This is the first activity launched and it redirects you to other activities: CharacterCreationActivity, CreateWordActivity, TestTagDbActivity, BrowseCharactersActivity, BrowseWordsActivity, BrowseLessonsActivity. The activity displays a ListView and has an OnItemClickListener that determines which activity to launch based on the item/CharSequence value chosen.

SCREENSHOT 2 & 3

Activity Class: CharacterCreationActivity

Helper class: DbAdapter

This activity handles creating and displaying a character.During character creation, the activity displays a CharacterCreationPane wrapped in a SquareLayout (to enforce squareness).

When the user switches modes, the activity switches a different pane into view. The animated playback is handled by a CharacterDisplayPane, and the tracing is handled by a CharacterTracePane (Note: when setting the template for CharacterTracePane, use the setTemplate() method not the setCharacterMethod(), the setCharacterMethod() is part of the inherited CharacterCreationPane.) When the user selects the save button, the activity sends the LessonCharacter to the database to be saved, and launches the TagActivity and passing it the characters id.

SCREENSHOT 4
Activity Class: TagActivity
Helper class: DbAdapter
This activity handles the tagging of words, characters, and lessons. It is passed an enumeration of the type of object it is tagging and the id of that item. The tags are placed in an ArrayAdapter which is set as the list view. When you add a new tag and click 'Add', onAddTagButtonClick() is called, grabbing the tag inputted into the EditText and then calling on DbAdapter.createTags/ createWordTags/createLessonTags dependong on what type the Intent passed in. The same goes for adding a private tag, except that the DbAdapter.updatePrivateTag is called instead.

SCREENSHOT 5
Activity Class: CreateWordActivity
Helper class: DbAdapter, LessonItemListAdapter
This activity handles the creation of words. A list view of all characters in the database are shown. Selecting a character adds a bitmap to a gallery at the bottom of the screen. Clicking 'Save Word' calls on the DbAdapter variable to add the word represented by the gallery into the database. Then a popup shows up (Screenshot 6). Clicking 'Add Tag' checks if the word was saved (there is a saved boolean variable) before it calls on TagActivity to launch (Screenshot 4)

SCREENSHOT 6
Activity Class: CreateWordActivity
Helper class: DbAdapter, LessonItemListAdapter
This is a popup window that handles adding a word to a new or existing lesson. The lessons are in a ListView with the use of LessonItemListAdapter, but there is also an EditText and Button that allows the user to create a new lesson there.

SCREENSHOT 7
Activity Class: TestTagDbActivity
Helper class: DbAdapter
This activity handles searching a character. The characters are in a ListView with the use of LessonItemListAdapter so that the image of the traced character is shown. Typing into the EditText and clicking Search calls DbAdapter.getCharts(input into the EditText). Clicking on a character launches CharacterCreationActivity.

SCREENSHOT 8
Activity Class: BrowseCharactersActivity
Helper class: DbAdapter, LessonItemListAdapter
This activity shows all the characters that have been created. The characters are

arranged in a listview using LessonItemListAdapter. When the user clicks on a character, CharacterCreationActivity is launched and the id of the character is passed in the Intent and also that the mode is “display”. This is through the clickOnItem() method that is called within onListItemClick(). The user can click and hold on a particular character, which calls onCreateContextMenu(). If you want to add tags, then it launches TagActivity with the character ID and the type “CHARACTER” passed in the intent. If you want to delete the character, then the DbAdapter calls on deleteCharacter(id), where the id is the character id. The showToast() method is called after either context menu item is selected, and the showToast() method simply displays the message of whether what was selected was successful or not.

SCREENSHOT 9

Activity Class: BrowseWordsActivity

Helper class: DbAdapter, LessonItemListAdapter

This activity shows all the words that have been created. It takes in the id of a lesson in which it will display. If no id is supplied (the case when you enter this activity from the main screen), it will display all words created.

This activity shows the words that are in the particular lesson selected. The words are arranged in a listview using LessonItemListAdapter. When the user clicks on a word, PhrasePracticeActivity is launched and the id of the word is passed in the Intent and also that the mode is “display”. This is through the clickOnItem() method that is called within onListItemClick(). The user can click and hold on a particular character, which calls onCreateContextMenu(). If you want to add the word to a new collection, then initiatePopupWindow() is called. This method displays a ListView of all existing lessons. When you select a lesson, DbAdapter.addWordToLesson method is called and showToast() displays a popup of whether it was successful or not. If you want to delete a word, DbAdapter.deleteWord(id), where id is the word id, is called and showToast(message) brings a popup with whether deletion was successful or not.

SCREENSHOT 10

Activity Class: PhrasePracticeActivity

This activity handles practicing and playing back a phrase of word. The list of characters at the top of the page is a gallery of bitmaps of the characters in the phrase. These bitmaps are created using the BitmapFactory class. The page uses a ViewAnimator to track all of the character panes, when the user changes between modes, the activity swaps between CharacterDisplayPanes and CharacterTracePanes being stored in the ViewAnimator (this is done in a somewhat hacky fashion, and can be done more elegantly). Most of the rest of the functionality behaves the same as the CharacterCreationActivity (and some of the code is taken from there).

SCREENSHOT 11

Activity Class: BrowseLessonsActivity

Helper class: DbAdapter, LessonListAdapter

This activity lists all lessons that the user has created. The Lesson objects are retrieved by the DbAdapter.getLessonById after grabbing all of the lesson id’s in the database. These Lesson objects are placed in LessonListAdapter. When the user clicks on a ListView item, the BrowseWordsActivity is launched to see the phrases for the lesson selected. The selected

Lesson object's id is passed into an Intent object and the BrowseWordsActivity is launched.

SCREENSHOT 12
Activity Class: BrowseWordsActivity
Helper class: DbAdapter, LessonItemListAdapter
This activity shows the words that are in the particular lesson selected. The words are arranged in a listview using LessonItemListAdapter. When the user clicks on a word, PhrasePracticeActivity is launched and the id of the word is passed in the Intent and also that the mode is "display". This is through the clickOnItem() method that is called within onListItemClick(). The user can click and hold on a particular character, which calls onCreateContextMenu(). If you want to add the word to a new collection, then initiatePopupWindow() is called. This method displays a ListView of all existing lessons. When you select a lesson, DbAdapter.addWordToLesson method is called and showToast() displays a popup of whether it was successful or not. If you want to delete a word, DbAdapter.deleteWord(id), where id is the word id, is called and showToast(message) brings a popup with whether deletion was successful or not.

Shared Helper Methods

DbAdapter

This is a helper method used by the majority of the activities that allows for access to the database. It handles all requests for data from the UI.

LessonItemListAdapter

This is the list layout for characters or words. It displays bitmap(s) on the left and private/public tags on the right.

Database Schema

For organizational purposes, we decide to put the database into 3NF in order to minimize redundancy and dependency.
Data Description Language for the database (with descriptions and justifications following):

```
CREATE TABLE Character (
        _id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT);
```

The Character table is used to hold all the information unique to one character. The only information held in this table is the unique identifier for the character and its private tag "name."

```
CREATE TABLE CharacterTag (
        _id INTEGER,
        tag TEXT NOT NULL,
        FOREIGN KEY(_id) REFERENCES Character(_id));
```

The CharacterTag table holds the public tags for each character. Each character can have multiple tags.

```
CREATE TABLE CharacterDetails (_id INTEGER PRIMARY KEY AUTOINCREMENT,
    CharId INTEGER,
    Stroke INTEGER NOT NULL,
    PointX DOUBLE NOT NULL,
    PointY DOUBLE NOT NULL,
    OrderPoint INTEGER NOT NULL,
    FOREIGN KEY(CharId) REFERENCES Character(_id));
```

The CharacterDetails table stores information for the strokes for each character. CharId refers to the character. Stroke is the stroke number for that character. PointX and PointY are coordinates for a given point on that stroke. OrderPoint is the order of that point for the given stroke.

```
CREATE TABLE Words (
    _id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT);
```

The Words table holds a unique identifier for the word and the name associated with that word, or the private tag.

```
CREATE TABLE WordsDetails (
    _id INTEGER,
    CharId INTEGER,
    WordOrder INTEGER NOT NULL,
    FlagUserCreated INTEGER,
    FOREIGN KEY(CharId) REFERENCES Character(_id),
    FOREIGN KEY(_id) REFERENCES Words(_id));
```

The WordsDetails table hold the characters that consist of each word. The CharId field refers to one of the Characters that has been created. WordOrder is the ordering of that character for the given word. FlagUserCreated is a field to determine whether or not the word was user created or auto-created (characters are auto-created as single character words upon creation).

```
CREATE TABLE WordsTag (
    _id INTEGER,
    tag TEXT NOT NULL,
    FOREIGN KEY(_id) REFERENCES Words(_id));
```

The WordsTag table stores the public tags associated with each word.

```
CREATE TABLE Lessons (
    _id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT);
```

The Lessons table stores each lesson with a unique id and name.

```
CREATE TABLE LessonsDetails (
        LessonId INTEGER,
        WordId INTEGER,
        LessonOrder INTEGER NOT NULL,
        FOREIGN KEY(LessonId) REFERENCES Lessons(_id),
        FOREIGN KEY(WordId) REFERENCES Words(_id));
```

The LessonsDetails table stores the words for each lesson. The LessonId field refers to a specific lesson. The WordId refers to a word in that lesson. LessonOrder is the order of that word in the lesson.

```
CREATE TABLE LessonTag (
        _id INTEGER,
        tag TEXT NOT NULL,
        FOREIGN KEY(_id) REFERENCES Lessons(_id));
```

The LessonTag table stores tags for each lesson.