**Gardening Application**

Joseph Podnar – Scrum Master / Project Owner

Wenchao Wang – Quality Assurance Manager

Glenn Selker - Software Developer

Muneeb Alvi - Software Developer

Charles Smith - Software Developer

June 24th, 2015

**CS 1530 SPRINT 2 DELIVERABLE**

**Sprint Accomplishments**

A lot was accomplished for sprint two. The team exceeded expectation with what they were able to create within the two weeks. Much of this success can be attributed to the three tiers of communication employed by the team. At the highest level, the whole group met for a three hour sprint planning meeting. Going in to the meeting, it was not planned that it would last as long as it did. However, after 10 minutes in front of a white board, it was clear that there was no unified vision for the product.

There were a lot of different opinions on the user interface. Some voices wanted the simplest possible user interface. Others wanted the most friendly user interface. Many people spoke their opinions passionately. In the end, a hybrid of many ideas was reached. For the few items where no compromise could be found, a vote was taken. Tasks were grouped into three categories: back-end, plant catalog, and garden map. Each of these tasks was championed by the three software developers.

The second level of communication was ten to fifteen minute standups following each software engineering lecture. Each of these meetings began with show and tell. Accomplishments were demonstrated and explained. It was clear that each software developer had been working hard, and they were excited to demonstrate the state of their tasks. This excitement helped propel constant improvement of the product during this sprint. After show and tell, the group troubleshooted problems and talked about what they needed for further progression.

The final level of communication was constant digital chatting through the Google Hangouts chat interface. This utility allowed the team to exceed expectation. Deep discussions were held at all times of the day. Frequently concepts and problems were discussed at midnight or later. People who were not up for this could easily catch up by reading what they missed the following day. This 24/7 open dialog allowed questions to be answered in the most expedient way possible.

The greatest change in process occurred with a road block. The initial plan for displaying the garden layout was to use Google Maps API v2. This violated K.I.S.S principles, as we were trying to build a garden on a framework with the complexity of modeling the whole earth. While it would have been entirely possible to supply custom map tiles and custom shapes, it became clear that the scope of this solution would prevent significant progress in other important avenues of the customer’s requirements. In the sprint planning meeting Android’s Canvas graphics resource was selected as an ideal replacement to Google Maps API v2.

A major improvement and change of direction with the application was the result of a customer interaction. The customer liked the idea of placing circles on a rectangle shaped garden, but they had a request. They wanted a few shapes to drag onto the garden bed. That way they could try and mock up a rough copy of what their garden actually looked like. They sent photos of their garden, and a drawing of the layout drafted by a landscape architect. These resources allowed us to do far better than offer a few shapes to mock up her layout. Using the drawing as a reference, an attractive digital representation of the garden was generated in Adobe Photoshop. This drawing now makes up the background of the digital garden layout.

**Sprint Challenges**

Programming a user interface is a new experience for me as I am employed as a C programmer working on a backend project. I found that user interfaces are tricky to write tests for, although testing is very necessary. I also found that it was a little like trial and error trying to get the graphical components to interact correctly. Also, I learned that testing on horizontal layouts, tablets, and real devices, not just emulators is necessary for my role on this team. My code could have been better had I tested it on a wider variety of devices while crafting the user interface.

- Glenn Selker

I was unable to install Intel’s **H**ardware **A**ccelerated E**x**ecution **M**anager due to the operating system on my computer and processor within. While emulation is possible without HAXM, the performance hit was too much to reasonably emulate the application. Due to this issue I was assigned backend requirements that were platform independent. I developed using Netbeans; other team members fixed my code’s syntax so that it would work in android studio. It was tough to get used to TDD. I was tempted to write code first and then test. It was hard to set up the JUnit tests for the code because this was my first time using it. Initially, I didn’t know how to compose tests until I realized that I should test each function based on simulated inputs and expected outputs.

- Muneeb Alvi

There were two large challenges with this sprint, the first being learning the android API. The android API is absolutely huge with many different ways to do the same thing. Luckily android has a strong online community with many solutions to the common problems people face. The second challenge would be our integration. For this sprint, many parts of code relied on other parts of code, this made putting the pieces together a little difficult. Git conflicts when merging were annoying to sort out. In the end, it was all cleanly merged.

-Charles Smith

**Completed User Stories**

As a user

I want to be greeted with a home screen on the Main activity

So I feel welcomed to the application

Difficulty: 2

As a user

I want a View garden button on main activity

So that I can choose where I want to start when I enter the application

Difficulty: 1

As a user

I want a View species button on main activity

So that I can choose where I want to start when I enter the application

Difficulty: 1

As a user

I want to see the ViewSpecies activity

So that I can choose a species to view

Difficulty: 4

As a user

I would like a background bitmap of my garden

So that I can place plants relative to my actual garden

Difficulty: 4

As a user

I would like the garden to move when I drag my finger across the screen

So that I can navigate around my garden

Difficulty: 8

As a user

I want to place a circle where I click

So that I can select a position for my plant

Difficulty: 4

As a user

I want clicking a new place and have the current circle move to this place

So that I can refine or move where I’m placing a circle

Difficulty: 4

**Garden Application Code**

The code for this project can be located at the following address:

https://github.com/cs1530-group1/garden\_app/tree/master

**User Story Choice**

It was important to get the team focused on individual parts of the project. Since there were many elements to create, parallel progress enabled multiple parts of the project to be developed asynchronously. The two main priorities of this build were to draw circles on the screen and display a clickable list of items. These can be considered the muscles we added to the walking skeleton. With these two parts the product could have a clear and demonstrable vision.

Creating and deleting plants and species was not necessary for this goal. Those user stories will be left for sprint three. Most of the remaining user stories are related to drawing circles and items on the clickable list. As such, it would be relatively impossible to complete them without at least those two features implemented.

The team had to learn a lot about the platform before they could dive in. This necessary research phase limited the actual coding time available to the software developers. Further time was needed to give functional code to the Quality Assurance Manager. He needed time to find issues so that they could be rectified before merging branches with the master. With these necessary restrictions the goals for this sprint were ambitious. They were also accomplished with flying colors. This was especially true of the digital garden layout. The initial work was completed quickly. This left time to add dragging support for moving the garden, and creating a background image for the garden to grow upon.

**Quality Assurance**

# **Emulator Device Specification:**

Resolution: 1280x800

CPU: Intel Atom (x86\_64)

RAM: 1536MB

Our client uses a Samsung Tab 4 – so those specs are geared towards her actual device. As for this sprint, the app only has to run on this device with no significant defects.

# **Defects in Garden App**

Garden App is the front end of the app, so most of the tests are done manually in Android emulator.

### The toast popups are not very responsive

**Status**: Closed

**Reproducibility**: Always

**Priority**: Low

**Description**: Found by tapping rapidly on different plant buttons in View Species List. The popups are not very responsive as a user clicks the buttons. This is an issue with Android’s Toast.makeText. No fix needed – these toasts are just placeholders for features that will be implemented later.

### Species list view's OnClickListener not responding to taps correctly

**Status**: Fixed

**Reproducibility**: Always

**Priority**: High

**Description**: Found by clicking on the species buttons on Species List View. The buttons do not register taps. Fixed by refactoring OnClickListener and adding a statement that was missing.

### Buttons on home screen not showing in correct positions

**Status**: Fixed

**Reproducibility**: Always

**Priority**: Low

**Description**: Found by running the app on emulator. The buttons do not appear in the right places. Fixed by modifying main activity’s xml with better positioning params.

### Minimizing the app in View Garden causes the app to crash

**Status**: Open

**Reproducibility**: Always

**Priority**: Medium

**Description**: Found by minimizing the app when the app is in View Garden. Trying to bring the app back up will cause the app to stop. Not fixed yet – cause of crash is unknown. The group may need some time to look into it.

### The tap/drag handling code favors drag too much on a physical device

**Status**: Open

**Reproducibility**: Always

**Priority**: High

**Description**: Found by dragging on the garden view on a physical android device. On the emulator, it appears to work fine, but on a physical device, it is hard to place a circle because the tolerance for the finger press and finger release coordinate is so tight. Not fixed yet – this will take some time to fix.

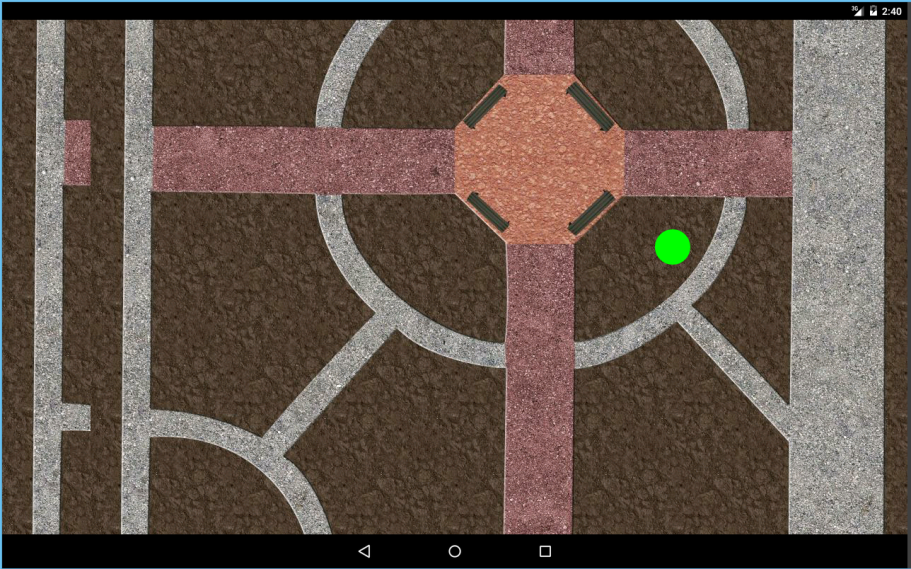
### Garden rendering issue on 1280x800 resolution

**Status**: Open

**Reproducibility**: Always

**Priority**: High

**Description**: Found by dragging on the garden view on emulator. As shown in the screenshot below, the left edge of the screen is rendered incorrectly. This is caused by garden background texture being smaller than the screen resolution. Not fixed yet – while this affects user experience, the app still functions. This needs to be fixed in the future.



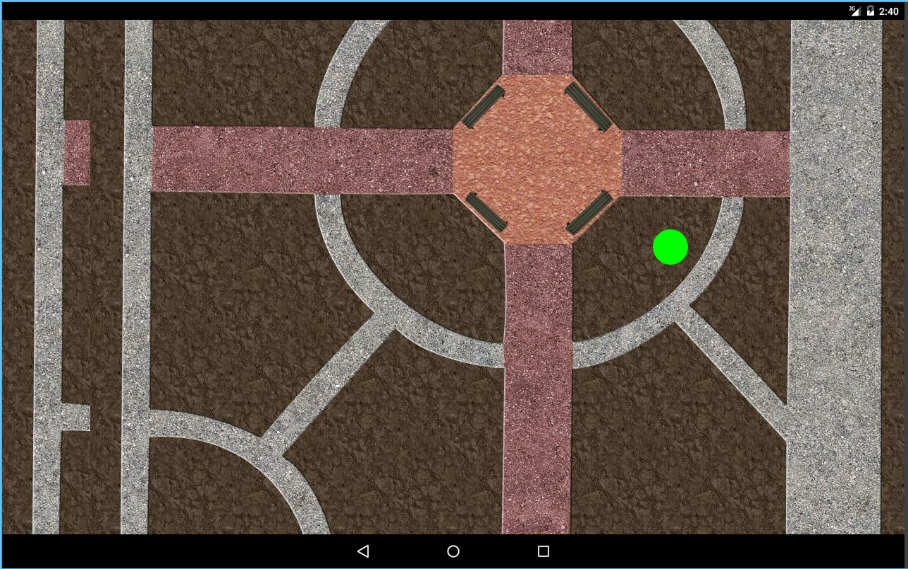
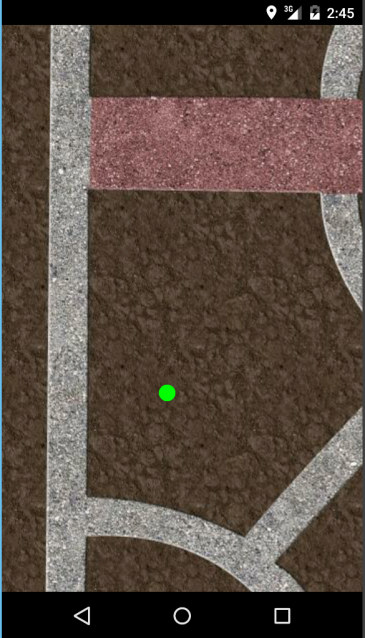
### Circles appears to be in different sizes on different screen resolutions

**Status**: Open

**Reproducibility**: Always

**Priority**: Low

**Description**: Found by tapping the screen on the garden view on emulator in different resolutions. As shown in the two screenshots below, the green circle is smaller on 1080x1920 than on 1280x800. Not fixed yet – the goal of this sprint is to get the app running on 1280x800, the resolution of the device our user plans to run this app on. Support for other resolutions is not currently prioritized.



# **Defects in Garden Engine**

Garden engine is tested with Unit Tests and code inspections.

### Garden serialize and deserialize use different formats

**Status**: Fixed

**Reproducibility**: Always

**Priority**: High

**Description**: Found by running garden unit tests. The GardenToString() produces a different text format than the one StringToGarden() uses to deserialize. Fixed by modifying the GardenToString and StringToGarden to use the same text format.

### Garden engine lacks documentation

**Status**: Open

**Reproducibility**: Always

**Priority**: High

**Description:** Found by code inspection. There is no JavaDoc documentation for any of the classes in garden engine. This should be prioritized and fixed as soon as possible in order to maintain the code base.

### Garden engine needs refactoring

**Status**: Open

**Reproducibility**: Always

**Priority**: High

**Description:** Found by code inspection. There are some design issues in Garden.java that need to be fixed.

* boolean StringToGarden(String garden) should be static Garden StringToGarden(String garden). This method should not change the state of an already created garden like it is currently doing. Instead, it should create a new garden.

This would enable the developer to construct a garden with  
 Garden myGarden = Garden.StringToGarden(gardenString);

For consistency it would also be a good idea to make GardenToString a static method, i.e. static String GardenToString(Garden garden)

* Add public getters and setters for the fields in Species and Plant. Also add JUnit tests for those.
* String[] ListSpecies() should be changed to String[] GetSpeciesNames() to avoid confusion. Do not enforce garden string formatting outside of this class.