Expectations

From CS164

Below are each milestone's expectations for each of the course's projects. All milestones are due at noon. See the course's syllabus (http://cdn.cs164.net/2012/spring/lectures/0/syllabus.pdf) (or the home page's Google Calendar) for each milestone's due date.

Contents

Project 0

Here's what to do for each milestone for Project 0.

Proposal

1. Decide who your partner will be this term.
2. Read the entirety of Project 0's spec (http://cdn.cs164.net/2012/spring/projects/0/project0.pdf) . Do the entire **Getting Started** section. Remember which of you is Alice.
3. Have a conversation with your partner about who will do what.
4. Visit https://bitbucket.org/alice/project0, where **alice** is the actual Bitbucket username of whoever was Alice while **Getting Started**.
5. Click **Wiki**.
6. Click **New**.
7. Input a **Page name** of **Proposal**, then click **Create**.
8. In the text area to the right of **Data**, summarize in a few sentences who will do what. (It's fine if you change your plans later.)
9. To the right of **Message**, input **Initial commit** or similar.
10. Click **Save**.
11. Click **Wiki** again to return to your wiki's **Home** page.
12. Click **Edit**.
13. In the text area to the right of **Data**, replace the default text with, minimally, a link to your app's proposal, as with this markup:

```
* [[Proposal]]
```

14. Submit this form (https://docs.google.com/spreadsheet/viewform?formkey=dG1EU3gwWHpuazZYcWlxdkxPWTVhRlE6MQ) . We'll then assign you and your partner a TF by the course's third week.

Design Doc, Style Guide

Design Doc

The format of your design doc is up to you, but it should somehow capture any and all design decisions that you and your partner make before (or while) implementing your app. Reasonable to incude in your design doc might be:

- list of database tables and fields (and their types) that you've decided to implement
- list of classes and methods (and their return types and/or arguments) that you've decided to implement
- photos of whiteboard drawings
- sketches of UIs
- summaries of features

When ready to create your design doc:

1. Visit https://bitbucket.org/alice/project0, where **alice** is the actual Bitbucket username of whoever was Alice while **Getting Started**.
2. Click **Wiki**.
3. Click **New**.
4. Input a **Page name** of **Design Doc**, then click **Create**.

5. In the text area to the right of **Data**, create your design doc. (It's fine if you decide to make changes to it later.) If, while developing your app, you find yourself constantly asking or answering questions with your partner, you will have failed to create a sufficiently thorough design doc!
6. To the right of **Message**, input **Initial commit** or similar.
7. Click **Save**.
8. Click **Wiki** again to return to your wiki's **Home** page.
9. Click **Edit**.
10. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Design Doc]]
```

If you'd like to include images in your design doc, see http://confluence.atlassian.com/display/BITBUCKET/Adding+Images+to+a+Wiki+Page and http://www.wikicreole.org/wiki/Creole1.0#section-Creole1.0-ImageInline.

Style Guide

The format of your style guide is up to you, but it should somehow capture the style conventions to which you and your partner will adhere for this app's:

- CSS declarations
- HTML tags
- JavaScript code
- PHP code
- SQL queries, if any

Before writing your own, review a few style guides to get a sense of their content:

- CS50 Style Guide (https://manual.cs50.net/Style_Guide)
- Google JavaScript Style Guide (http://google-styleguide.googlecode.com/svn/trunk/javascriptguide.xml)
- PEAR Coding Standards (http://pear.php.net/manual/en/standards.php)
- Zend Framework Coding Standard for PHP (http://framework.zend.com/manual/en/coding-standard.html)

When ready to create your style guide:

1. Have a (heated) conversation with your partner and decide on your style conventions.
2. Visit https://bitbucket.org/alice/project0, where **alice** is the actual Bitbucket username of whoever was Alice while **Getting Started**.
3. Click **Wiki**.
4. Click **New**.
5. Input a **Page name** of **Style Guide**, then click **Create**.
6. In the text area to the right of **Data**, create your style guide. (It's fine if you decide to make changes to it later.) If you ultimately submit code whose style is not consistent throughout, you will have failed to create (or adhere to) a sufficiently thorough style guide!
7. To the right of **Message**, input **Initial commit** or similar.
8. Click **Save**.
9. Click **Wiki** again to return to your wiki's **Home** page.
10. Click **Edit**.
11. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Style Guide]]
```

Beta

As with any beta (http://en.wikipedia.org/wiki/Software_release_life_cycle#Beta) , your project's beta should be feature-complete (http://en.wikipedia.org/wiki/Feature_complete) , even though it may have some bugs. Per the syllabus (http://cdn.cs164.net/2012/spring/lectures/0/syllabus.pdf) , it will be evaluated primarily along axes of scope, design, and style (whereas your project's release will be evaluated primarily along those axes as well as correctness).

When ready to submit your beta, only one member of your two-person team should follow the steps below. Let's call the team member who'll be submitting "Alice" (and the team member who won't be submitting "Bob"). For the purposes of submitting, it doesn't matter which of you are which: even if you were Alice for your Design Doc and Style Guide, you can still be Bob this time; and vice versa.

1. Ensure that you (Alice) have the latest version of everything in `~/vhosts/project0`, including, if applicable, your MySQL database. If Bob has a newer version of a MySQL database than you, see https://www.cs164.net/Hooks#MySQL for instructions on how to add and push it to your Bitbucket repository, so that you can pull it down. **If you based your own `pre-commit` hook on those instructions prior to 11:30pm on Thu 2/9, re-visit the page for an updated hook (whose dump no longer hardcodes your database's name).**
2. Create a text file called `README` in `~/vhosts/project0` that explains, in succinct steps, how your TF and classmates can get your project up and running in their own `~/vhosts` directory. Odds are those instructions will resemble the below, where `alice` is your actual Bitbucket username:

```
# clone repo into ~/vhosts/alice
cd ~/vhosts
git clone git@bitbucket.org:alice/project0.git alice

# chmod all directories 711
find ~/vhosts/alice -type d -exec chmod 711 {} \;

# chmod all PHP files 600
find ~/vhosts/alice -type f -name *.php -exec chmod 600 {} \;

# chmod most everything else 644
find ~/vhosts/alice -type f \( -name *.css -o -name *.gif -o -name *.html -o -name *.js -o -na

# create a MySQL database for project
mysql -u jharvard -p -e 'CREATE DATABASE jharvard_alice'

# import SQL dump into database
mysql -u jharvard -p jharvard_alice < ~/vhosts/alice/mysql/jharvard_project0.sql

# change the value of $db['default']['database'] to be 'jharvard_alice'
gedit ~/vhosts/alice/application/config/database.php

# append '127.0.0.1 alice' to /etc/hosts
sudo gedit /etc/hosts
```

Note that these instructions assume that you've dumped your MySQL database, if any, per https://www.cs164.net/Hooks#MySQL. **If you based your own `pre-commit` hook on those instructions prior to 11:30pm on Thu 2/9, re-visit the page for an updated hook (whose dump no longer hardcodes your database's name).**

3. Add the `README` to your Bitbucket repo, as with:

```
git add README
git commit -m 'Added README'
git push
```

4. Confirm that your `README` is now displayed at the bottom of https://bitbucket.org/alice/project0, where **alice** is your actual Bitbucket username.
5. Ask Bob to follow your `README`'s instructions inside of his own appliance, to ensure they're 100% correct. If he's able to get your project up and running at http://alice/, where **alice** is still your actual Bitbucket username, in addition to his own copy that's likely still http://project0/, odds are your `README`'s instructions are correct! If incorrect, though, make (and push) any changes to your `README` as needed.
6. "Tag" your official submission as follows:

```
cd ~/vhosts/project0
git tag --force beta
git push --tags
```

7. Look up your TF's Bitbucket username at https://www.cs164.net/Staff. Then visit https://bitbucket.org/alice/project0, where **alice** is your actual Bitbucket username. Click the **Admin** tab at top, then click **Access management** at left. In the text field under **Users**, input your TF's Bitbucket username, then click Admin at right.

That's it! If you make any changes to your project and want to re-submit (before the beta's deadline), odds are you'll want to execute, at least, these commands after making those changes:

```
cd ~/vhosts/project0
git add --all
git commit -m "Last-minute changes"
git push
git tag --force beta
git push --tags
```

Code Reviews

For Project 0, you don't need to review classmates' code just yet, but your TF will review your code! You should receive feedback on your beta from your TF by Wed 2/15. If you don't, email your TF and CC heads@cs164.net (mailto:heads@cs164.net) to inquire.

Even before receiving that feedback, you should dive back into your project, completing any features you didn't complete in time for your beta. Upon receiving that feedback, you should dive in again, improving your project's correctness, design, and style, in time for your project's release (*i.e.*, final submission) on Mon 2/20 by noon.

If in need of a hand or some counsel, know that there will be opportunities for code reviews:

- Tue 2/14, 6pm – 8pm, Pierce 301
- Wed 2/15, 6:00pm – 8pm, Pierce 301
- Thu 2/16, 6pm – 8pm, Pierce 301

And there will be a brand-new (filmed) section:

- Wed 2/15, 4pm – 6:00pm, Pierce 301

Release

To submit your release, only you or your partner (not both) need to follow these steps:

1. Ensure that you have the latest version of your code (and your partner's code) in `~/vhosts/project0`, as via `git pull`.
2. Ensure that you have a `README` in `~/vhosts/project0` (which you should have created prior to submission of your Beta). Update that `README` with any thoughts you'd like to communicate to your TF. For instance, if you made some design decision that you're worried might be questioned, argue your case. If you made some design decision that you know is suboptimal but you had your reasons (*e.g.*, difficulty getting an alternative design to work), explain as much. (Merely disclaiming suboptimal designs won't necessary yield a higher score, but we'll at least know you gave the matter some thought and didn't make a suboptimal decision in a vacuum.)
3. Tag and push your code for submission as follows:

```
cd ~/vhosts/project0
git add --all
git commit -m "Ready for release"
git push
git tag --force release
git push --tags
```

4. Ask your partner to pull down your code via `git` and confirm that everything is indeed in working order, lest your TF find otherwise.
5. Profit!

Project 1

Here's what to do for each milestone for Project 1.

Proposal

1. Read the entirety of Project 1's spec (http://cdn.cs164.net/2012/spring/projects/1/project1.pdf) . Do the entire **Getting Started** section. Remember which of you is Alice.
2. Have a conversation with your partner about what you will do for your choice of mobile web apps and who will do what.
3. Visit https://bitbucket.org/alice/project1, where **alice** is the actual Bitbucket username of whoever was Alice while **Getting Started**.
4. Click **Wiki**.
5. Click **New**.
6. Input a **Page name** of **Proposal**, then click **Create**.
7. In the text area to the right of **Data**, propose a project, formatting your proposal as follows:

```
== Title ==

your project's title
```

```
== Summary ==

a sentence summarizing your project

== Features ==

=== Alpha ===

bulleted list of features that will be in your alpha

=== Release ===

bulleted list of features that will be in your release

== Implementation Details ==

bulleted list of any frameworks, languages, libraries, or other technologies with which you'll

== Ownership ==

bulleted lists of who will do what
```

8. To the right of **Message**, input **Initial commit** or similar.
9. Click **Save**.
10. Click **Wiki** again to return to your wiki's **Home** page.
11. Click **Edit**.
12. In the text area to the right of **Data**, replace the default text with, minimally, a link to your app's proposal, as with this markup:

```
* [[Proposal]]
```

Design Doc, Style Guide

Design Doc

The format of your design doc is up to you, but it should somehow capture any and all design decisions that you and your partner make before (or while) implementing your app. Reasonable to incude in your design doc might be:

- list of database tables and fields (and their types) that you've decided to implement
- list of classes and methods (and their return types and/or arguments) that you've decided to implement
- photos of whiteboard drawings
- sketches of UIs
- summaries of features

When ready to create your design doc:

1. Visit https://bitbucket.org/alice/project1, where **alice** is the actual Bitbucket username of whoever was Alice while **Getting Started**.
2. Click **Wiki**.
3. Click **New**.
4. Input a **Page name** of **Design Doc**, then click **Create**.
5. In the text area to the right of **Data**, create your design doc. (It's fine if you decide to make changes to it later.) If, while developing your app, you find yourself constantly asking or answering questions with your partner, you will have failed to create a sufficiently thorough design doc!

6. To the right of **Message**, input **Initial commit** or similar.
7. Click **Save**.
8. Click **Wiki** again to return to your wiki's **Home** page.
9. Click **Edit**.
10. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Design Doc]]
```

If you'd like to include images in your design doc, see http://www.wikicreole.org/wiki/Creole1.0#section-Creole1.0-ImageInline and http://confluence.atlassian.com/display/BITBUCKET/Adding+Images+to+a+Wiki+Page and. Though you might find it easiest to upload images to imgur.com (http://imgur.com/) and then embed those.

Style Guide

The format of your style guide is up to you, but it should somehow capture the style conventions to which you and your partner will adhere for this app's:

- CSS declarations
- HTML tags
- JavaScript code, if any
- PHP code, if any
- SQL queries, if any

When ready to create your style guide:

1. Have a (heated) conversation with your partner and decide on your style conventions.
2. Visit https://bitbucket.org/alice/project1, where **alice** is the actual Bitbucket username of whoever was Alice while **Getting Started**.
3. Click **Wiki**.
4. Click **New**.
5. Input a **Page name** of **Style Guide**, then click **Create**.
6. In the text area to the right of **Data**, create your style guide. (It's fine if you decide to make changes to it later.) If you ultimately submit code whose style is not consistent throughout, you will have failed to create (or adhere to) a sufficiently thorough style guide!
7. To the right of **Message**, input **Initial commit** or similar.
8. Click **Save**.
9. Click **Wiki** again to return to your wiki's **Home** page.
10. Click **Edit**.
11. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Style Guide]]
```

It's fine to copy/paste from Project 0's Style Guide as needed, but take care to incorporate any feedback from your TF.

Alpha

To submit your alpha, only you or your partner (not both) need to follow these steps:

1. Ensure that you have the latest version of your code (and your partner's code) in `~/vhosts/project1/`, as via `git pull`.
2. Ensure that you have a `README` in `~/vhosts/project1/` (which you should have created prior to submission of your Alpha). Update that `README` with any thoughts you'd like to communicate to your TF. For instance, if you made some design decision that you're worried might be questioned, argue your case. If you made some design decision that you know is suboptimal but you had your reasons (*e.g.*, difficulty getting an alternative design to work), explain as much. (Merely disclaiming suboptimal designs won't necessary yield a higher score, but we'll at least know you gave the matter some thought and didn't make a suboptimal decision in a vacuum.)
3. Tag and push your code for submission as follows:

```
cd ~/vhosts/project1/
git add --all
git commit -m "alpha version"
git push
git tag --force alpha
git push --tags
```

4. Ask your partner to pull down your code via `git` and confirm that everything is indeed in working order, lest your TF find otherwise.
5. Profit!

Release

To submit your release, only you or your partner (not both) need to follow these steps:

1. Ensure that you have the latest version of your code (and your partner's code) in `~/vhosts/project1/`, as via `git pull`.
2. Ensure that you have a `README` in `~/vhosts/project1/` that includes any thoughts you'd like to communicate to your TF. For instance, if you made some design decision that you're worried might be questioned, argue your case. If you made some design decision that you know is suboptimal but you had your reasons (*e.g.*, difficulty getting an alternative design to work), explain as much. (Merely disclaiming suboptimal designs won't necessary yield a higher score, but we'll at least know you gave the matter some thought and didn't make a suboptimal decision in a vacuum.)
3. Tag and push your code for submission as follows:

```
cd ~/vhosts/project1/
git add --all
git commit -m "Ready for release"
git push
git tag --force release
git push --tags
```

4. Ask your partner to pull down your code via `git` and confirm that everything is indeed in working order, lest your TF find otherwise.
5. Profit!

Project 2

Here's what to do for each milestone for Project 2.

Proposal

1. Submit this form (https://docs.google.com/spreadsheet/viewform?formkey=dFBkMHIwLTZxZlktLUZROHpaNHZ2N2c6MQ) .
2. Read the entirety of Project 2's spec (http://cdn.cs164.net/2012/spring/projects/2/project2.pdf) . Do the entire **Bitbucket** section. Remember which of you is Alice.
3. Have a conversation with your partner about who will do what.
4. Visit https://bitbucket.org/alice/project2, where **alice** is the actual Bitbucket username of whoever was Alice.
5. Click **Wiki**.
6. Click **New**.
7. Input a **Page name** of **Proposal**, then click **Create**.
8. In the text area to the right of **Data**, summarize in a few sentences who will do what. (It's fine if you change your plans later.)
9. To the right of **Message**, input **Initial commit** or similar.
10. Click **Save**.
11. Click **Wiki** again to return to your wiki's **Home** page.
12. Click **Edit**.
13. In the text area to the right of **Data**, replace the default text with, minimally, a link to your app's proposal, as with this markup:

```
* [[Proposal]]
```

Design Doc, Style Guide

Design Doc

The format of your design doc is up to you, but it should somehow capture any and all design decisions that you and your partner make before (or while) implementing your app. Reasonable to include in your design doc might be:

- list of classes and methods (and their return types and/or arguments) that you've decided to implement
- photos of whiteboard drawings
- sketches of UIs
- summaries of features

When ready to create your design doc:

1. Visit https://bitbucket.org/alice/project2, where **alice** is the actual Bitbucket username of whoever was Alice.
2. Click **Wiki**.
3. Click **New**.
4. Input a **Page name** of **Design Doc**, then click **Create**.
5. In the text area to the right of **Data**, create your design doc. (It's fine if you decide to make changes to it later.) If, while developing your app, you find yourself constantly asking or answering questions with your partner, you will have failed to create a sufficiently thorough design doc!

6. To the right of **Message**, input **Initial commit** or similar.
7. Click **Save**.
8. Click **Wiki** again to return to your wiki's **Home** page.
9. Click **Edit**.
10. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Design Doc]]
```

If you'd like to include images in your design doc, see http://confluence.atlassian.com/display/BITBUCKET/Adding+Images+to+a+Wiki+Page and http://www.wikicreole.org/wiki/Creole1.0#section-Creole1.0-ImageInline.

Style Guide

The format of your style guide is up to you, but it should somehow capture the style conventions to which you and your partner will adhere for this app's:

- Objective-C code

When ready to create your style guide:

1. Have a (heated) conversation with your partner and decide on your style conventions.
2. Visit https://bitbucket.org/alice/project2, where **alice** is the actual Bitbucket username of whoever was Alice.
3. Click **Wiki**.
4. Click **New**.
5. Input a **Page name** of **Style Guide**, then click **Create**.
6. In the text area to the right of **Data**, create your style guide. (It's fine if you decide to make changes to it later.) If you ultimately submit code whose style is not consistent throughout, you will have failed to create (or adhere to) a sufficiently thorough style guide!
7. To the right of **Message**, input **Initial commit** or similar.
8. Click **Save**.
9. Click **Wiki** again to return to your wiki's **Home** page.
10. Click **Edit**.
11. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Style Guide]]
```

Alpha

To submit your alpha, only you or your partner (not both) need to follow these steps:

1. Ensure that you have the latest version of your code (and your partner's code) in `project2/`, as via **File > Source Code > Pull...** in Xcode or via `git pull`.
2. Create a `README` in `project2/` (using Xcode or any text editor) with any thoughts you'd like to communicate to your TF. For instance, if you made some design decision that you're worried might be questioned, argue your case. If you made some design decision that you know is suboptimal but you had your reasons (*e.g.*, difficulty getting an alternative design to work), explain as much. (Merely disclaiming suboptimal designs won't necessary yield a higher score, but we'll at least know you gave

the matter some thought and didn't make a suboptimal decision in a vacuum.)
3. Tag and push your code for submission in a Terminal window as follows, taking care to submit your top-level `project2/` directory (*i.e.*, `/path/to/project2/`, not `/path/to/project2/project2/`):

```
cd /path/to/project2/
git add --all
git commit -m "alpha version"
git push
git tag --force alpha
git push --tags
```

4. Ask your partner to pull down your code via `git` and confirm that everything is indeed in working order, lest your TF find otherwise.
5. Profit!

Release

To submit your release, only you or your partner (not both) need to follow these steps:

1. Ensure that you have the latest version of your code (and your partner's code) in `project2/`, as via **File > Source Code > Pull...** in Xcode or via `git pull`.
2. Ensure that you have a `README` in `project2/` that includes any thoughts you'd like to communicate to your TF. For instance, if you made some design decision that you're worried might be questioned, argue your case. If you made some design decision that you know is suboptimal but you had your reasons (*e.g.*, difficulty getting an alternative design to work), explain as much. (Merely disclaiming suboptimal designs won't necessary yield a higher score, but we'll at least know you gave the matter some thought and didn't make a suboptimal decision in a vacuum.)
3. Tag and push your code for submission in a Terminal window as follows, taking care to submit your top-level `project2/` directory (*i.e.*, `/path/to/project2/`, not `/path/to/project2/project2/`):

```
cd /path/to/project2/
git add --all
git commit -m "Ready for release"
git push
git tag --force release
git push --tags
```

4. Ask your partner to pull down your code via `git` and confirm that everything is indeed in working order, lest your TF find otherwise.
5. Profit!

Project 3

Here's what to do for each milestone for Project 3.

Proposal

1. Read the entirety of Project 3's spec (http://cdn.cs164.net/2012/spring/projects/3/project3.pdf) . Do the entire **Bitbucket** section. Remember which of you is Alice.

2. Have a conversation with your partner about who will do what.
3. Visit https://bitbucket.org/alice/project3, where **alice** is the actual Bitbucket username of whoever was Alice.
4. Click **Wiki**.
5. Click **New**.
6. Input a **Page name** of **Proposal**, then click **Create**.
7. In the text area to the right of **Data**, summarize in a few sentences who will do what. (It's fine if you change your plans later.)
8. To the right of **Message**, input **Initial commit** or similar.
9. Click **Save**.
10. Click **Wiki** again to return to your wiki's **Home** page.
11. Click **Edit**.
12. In the text area to the right of **Data**, replace the default text with, minimally, a link to your app's proposal, as with this markup:

```
* [[Proposal]]
```

Design Doc, Style Guide

Design Doc

The format of your design doc is up to you, but it should somehow capture any and all design decisions that you and your partner make before (or while) implementing your app. Reasonable to include in your design doc might be:

- list of classes and methods (and their return types and/or arguments) that you've decided to implement
- photos of whiteboard drawings
- sketches of UIs
- summaries of features

When ready to create your design doc:

1. Visit https://bitbucket.org/alice/project3, where **alice** is the actual Bitbucket username of whoever was Alice.
2. Click **Wiki**.
3. Click **New**.
4. Input a **Page name** of **Design Doc**, then click **Create**.
5. In the text area to the right of **Data**, create your design doc. (It's fine if you decide to make changes to it later.) If, while developing your app, you find yourself constantly asking or answering questions with your partner, you will have failed to create a sufficiently thorough design doc!
6. To the right of **Message**, input **Initial commit** or similar.
7. Click **Save**.
8. Click **Wiki** again to return to your wiki's **Home** page.
9. Click **Edit**.
10. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Design Doc]]
```

If you'd like to include images in your design doc, see

http://confluence.atlassian.com/display/BITBUCKET/Adding+Images+to+a+Wiki+Page and http://www.wikicreole.org/wiki/Creole1.0#section-Creole1.0-ImageInline.

Style Guide

The format of your style guide is up to you, but it should somehow capture the style conventions to which you and your partner will adhere for this app's:

- Objective-C code

When ready to create your style guide:

1. Have a (heated) conversation with your partner and decide on your style conventions.
2. Visit https://bitbucket.org/alice/project2, where **alice** is the actual Bitbucket username of whoever was Alice.
3. Click **Wiki**.
4. Click **New**.
5. Input a **Page name** of **Style Guide**, then click **Create**.
6. In the text area to the right of **Data**, create your style guide. **Assuming you'll adhere to the same style conventions as you did for Project 2, it suffices to provide a link to Project 2's Style Guide.**
7. To the right of **Message**, input **Initial commit** or similar.
8. Click **Save**.
9. Click **Wiki** again to return to your wiki's **Home** page.
10. Click **Edit**.
11. In the text area to the right of **Data**, add, minimally, a link to your app's style guide, as with this markup:

```
* [[Style Guide]]
```

Alpha

To submit your alpha, only you or your partner (not both) need to follow these steps:

1. Ensure that you have the latest version of your code (and your partner's code) in `project3/`, as via **File > Source Code > Pull...** in Xcode or via `git pull`.
2. Create a `README` in `project3/` (using Xcode or any text editor) with any thoughts you'd like to communicate to your TF. For instance, if you made some design decision that you're worried might be questioned, argue your case. If you made some design decision that you know is suboptimal but you had your reasons (*e.g.*, difficulty getting an alternative design to work), explain as much. (Merely disclaiming suboptimal designs won't necessary yield a higher score, but we'll at least know you gave the matter some thought and didn't make a suboptimal decision in a vacuum.)
3. Tag and push your code for submission in a Terminal window as follows, taking care to submit your top-level `project3/` directory (*i.e.*, `/path/to/project3/`, not `/path/to/project3/project3/`):

```
cd /path/to/project3/
git add --all
git commit -m "alpha version"
git push
git tag --force alpha
git push --tags
```

4. Ask your partner to pull down your code via `git` and confirm that everything is indeed in working order, lest your TF find otherwise.
5. Profit!

Release

First, both you AND your partner should submit this form (https://docs.google.com/spreadsheet/viewform?formkey=dDdNRWxyNGZxS3FGX3dmMTItZDNVZ3c6MA) .

Then, to submit your release, only you OR your partner (not both) need to follow these steps:

1. Ensure that you have the latest version of your code (and your partner's code) in `project3/`, as via **File > Source Code > Pull...** in Xcode or via `git pull`.
2. Ensure that you have a `README` in `project3/` that includes any thoughts you'd like to communicate to your TF. For instance, if you made some design decision that you're worried might be questioned, argue your case. If you made some design decision that you know is suboptimal but you had your reasons (*e.g.*, difficulty getting an alternative design to work), explain as much. (Merely disclaiming suboptimal designs won't necessary yield a higher score, but we'll at least know you gave the matter some thought and didn't make a suboptimal decision in a vacuum.)
3. Tag and push your code for submission in a Terminal window as follows, taking care to submit your top-level `project3/` directory (*i.e.*, `/path/to/project3/`, not `/path/to/project3/project3/`):

```
cd /path/to/project3/
git add --all
git commit -m "Ready for release"
git push
git tag --force release
git push --tags
```

4. Ask your partner to pull down your code via `git` and confirm that everything is indeed in working order, lest your TF find otherwise.
5. Profit!