General Assembly
MOB Project Brief

MID-COURSE PROJECT

Working solo or in pairs, build a functional clone of the iOS calculator app from scratch. Use the tools we've covered so far to build a template for your app, flesh out its UI and build its data models. In order to get exposure to technical speaking you will also have the option of giving a short technical presentation to the class on a number of preselected topics.

GOALS OF PROJECT

- Build an app from scratch, without an existing codebase
- Create a simple (no persistence, no networking) app with defined, familiar and expected behavior
- Build the end-to-end components required for a simple utility app: Models, views and controllers
- Communicate an app idea/technical architecture to a technical audience

PROJECT REQUIREMENTS

Your app must:

- Have a number display, which displays either the current input or the result of the desired calculation
    - Format: App contains screen numerical display that adjusts to button taps
- Have a number keypad that supports input of 0-9 and decimal points
    - Format: App contains screen with desired buttons that adjust calculator display accordingly
- Have an equals button that evaluates the desired calculation
    - Format: App contains screen with equals button
- Add, multiply, subtract, divide both positive and negative integers and floats
    - Format: App contains screen with desired buttons that work accordingly for all integers, floats and signs
- Have a 'percent' button
    - Format: App contains percent button which divides currently displayed number by 100
- Have 'clear' button
    - Format: App contains clear button that clears the current display
- Have a +/- button
    - Format: App contains screen with +/- button that inverses displayed number's sign
- Support portrait orientation
    - Format: App works in portrait mode, but does not necessarily work in landscape mode

BONUS OPPORTUNITIES

Your app can:

- Handle errors (e.g. divide by zero)
    - Format: App displays 'Error' when given an impossible calculation to compute
- Handle very large or small numbers (e.g. 1e+18)
    - Format: App displays extremely large and small numbers in order of magnitude format (e.g. 1e+20)
- Have a UI that closely matches the Apple calculator UI or another similar, aesthetically pleasing calculator app
    - Format: App UI is aesthetically pleasing
- Support memory functions (mc, m+, m-, mr)
    - Format: App contains memory function buttons that are functional
- Support some or all of the additional mathematical features seen in the iOS calculator's landscape mode (e.g. trig functions, parens, exponents, random numbers, etc)
    - Format: App contains some or all additional mathematical functions

You/your group can:

- Give a short presentation on a number of preselected topics during week 6.
    - Topics include:
        - The various ways to lay out views: auto layout constraints, stack views
        - Keeping track of values in their calculator: Where is your data stored, and how do you change it?
        - Mapping buttons to actions: Where does the action taken when you tap on a button live? How do you connect them?
    - Format: A <5 minute talk on how your team approached one of the above topics. Include discussion of possible alternate implementations, how they might work, and why yours is preferable.
-

DELIVERABLES

- Final app project (code, resources, project file, app description, app screenshots) posted on Github
- Optional presentation on technical topic

TIMELINE

DUE DATE	DELIVERABLE
2/1/2016	UX Completed, number buttons hooked up
2/8/2016	Arithmetic operations buttons hooked up and working.

SUGGESTED WAYS TO GET STARTED

- Start simple when it comes to UX. This app does not need to be a complete UI clone of the iOS calculator app, just a near functional clone.
- Start with researching how you can translate strings to numbers, and numbers to strings. What APIs will you need to use to display the numbers you want to display?
- Brainstorm what your model classes will look like, and how they will change. What do you need to store in your models?
- Once you've thought about all three, do your best to put all associated tasks on Trello. Try to be as specific as possible when specifying deliverables, especially if working in a team.

RESOURCES

- Trello (optional task tracking)
- iOS frameworks (frameworks)

EVALUATION

Your planning abilities, project, codebase and presentation will be evaluated using this rubric:

LINK TO RUBRIC

INSTRUCTOR NOTE:

- App Readiness, Execution & Scope are to be graded while using the submitted app
- Presentation & Communication are to be graded during student presentation of their app
- All Technical items are to be graded while reviewing the submitted code for the app