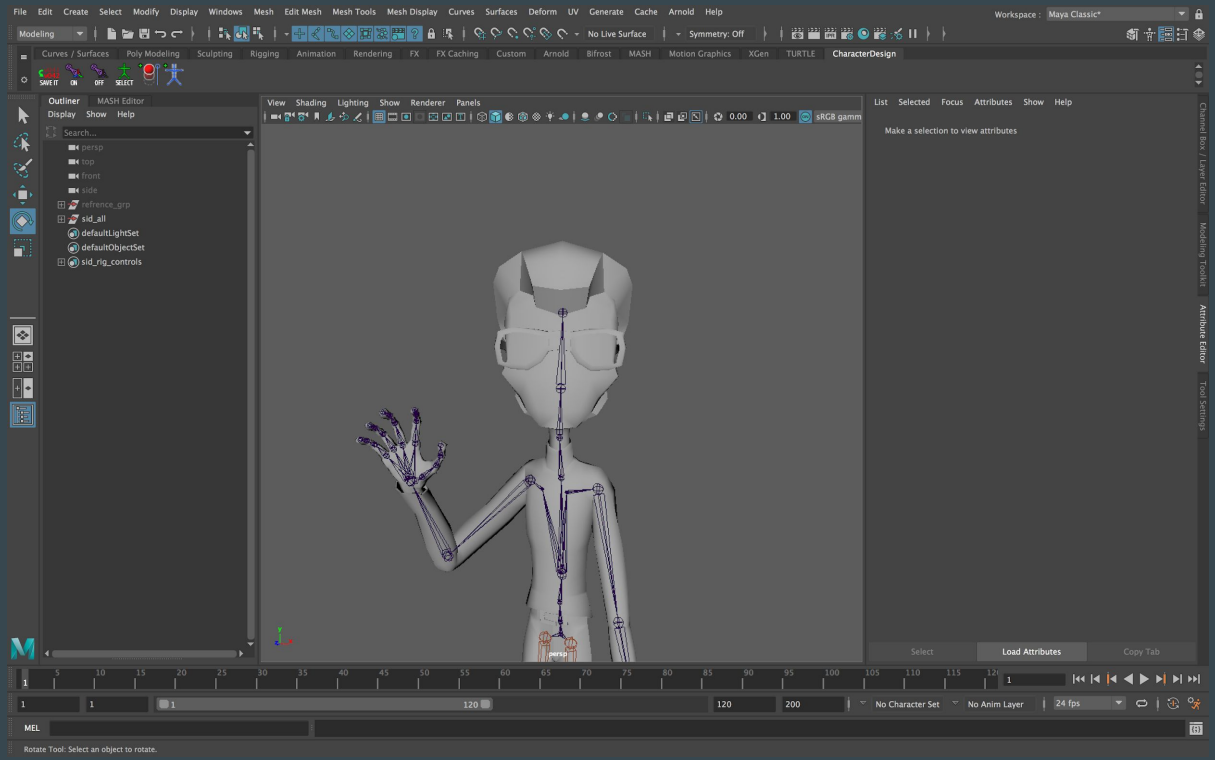
TyrantVC

Vlad Loyko, Carson Wilk, Ben Celsi, Annie Gesellchen

Motivation

What is Maya?

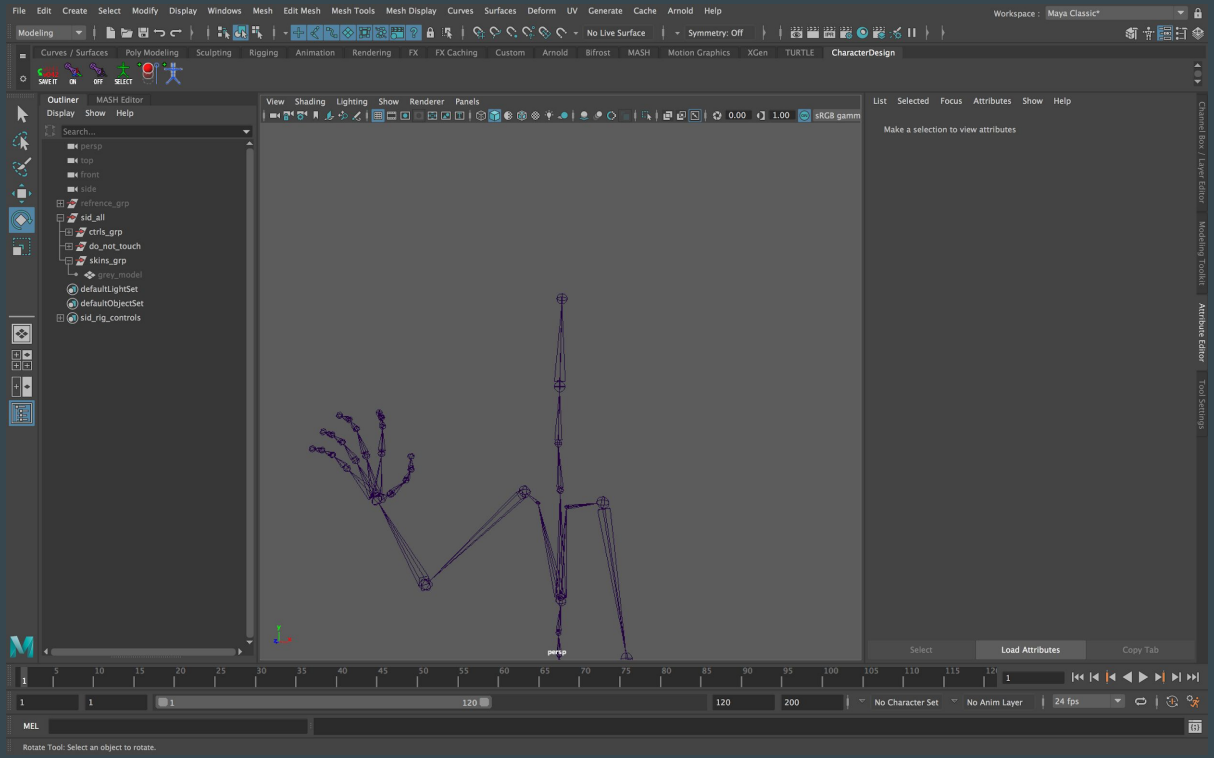
- Mainly a computer animation & modeling software
- Used by huge animation & gaming studios such as Pixar, ILM, Rockstar, etc.
- Main audience is artists and animators.



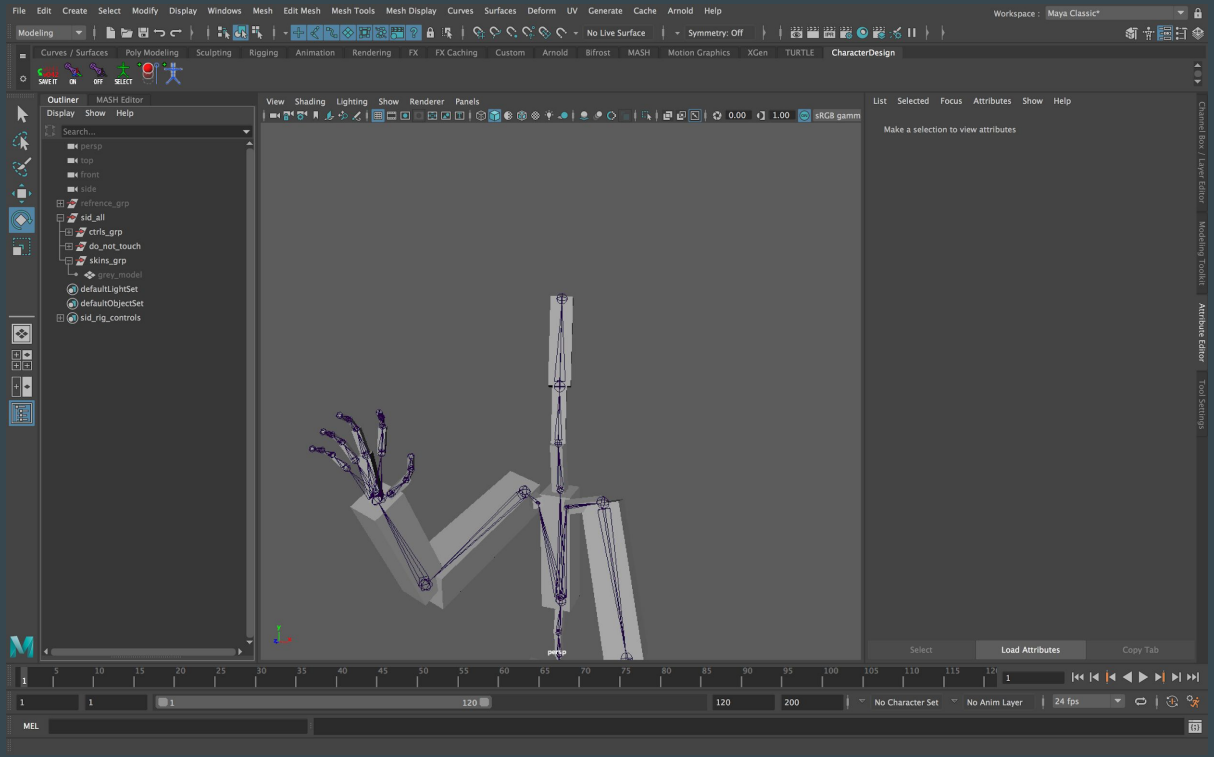
Why script in Maya?

- Automate/speed up complicated or repetitive tasks.
- Apply same behaviors over multiple Maya projects.
- Provide specialized tools for animators that do not exist in the core program.

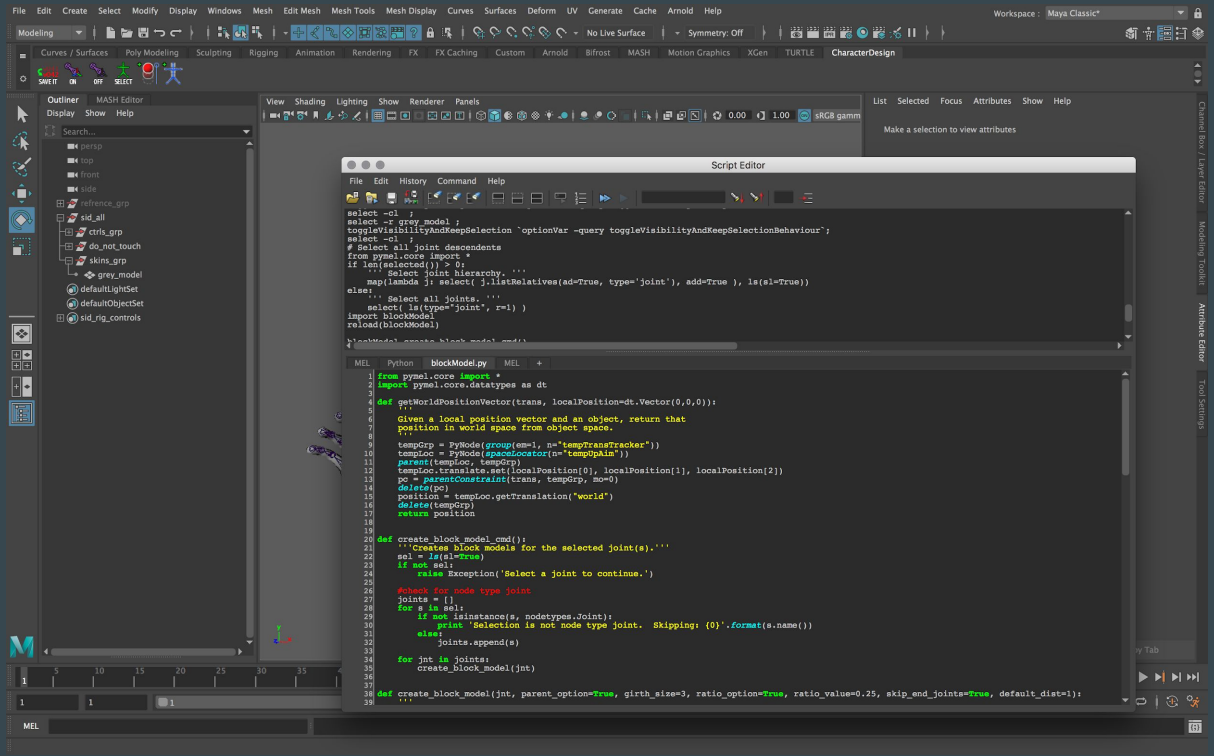
Example: Block Model Script



Example: Block Model Script



Example: Block Model Script



Scripting in Maya

- Maya Project files and scripts are completely separate and can exist in separate locations.
- Maya Script editor is only slightly better than a simple text editor.
- Writing helper plugins is essential when working on large projects.

Script Editor

File Edit History Command Help

```
select -cl  ;
select -r grey_model ;
toggleVisibilityAndKeepSelection `optionVar -query toggleVisibilityAndKeepSelectionBehaviour`;
select -cl  ;
# Select all joint descendents
from pymel.core import *
if len(selected()) > 0:
    ''' Select joint hierarchy. '''
    map(lambda j: select( j.listRelatives(ad=True, type='joint'), add=True ), ls(sl=True))
else:
    ''' Select all joints. '''
    select( ls(type="joint", r=1) )
import blockModel
reload(blockModel)
```

MEL Python blockModel.py MEL +

```
from pymel.core import *
import pymel.core.datatypes as dt

def getWorldPositionVector(trans, localPosition=dt.Vector(0,0,0)):
    '''
    Given a local position vector and an object, return that
    position in world space from object space.
    '''
    tempGrp = PyNode(group(em=1, n="tempTransTracker"))
    tempLoc = PyNode(spaceLocator(n="tempUpAim"))
    parent(tempLoc, tempGrp)
    tempLoc.translate.set(localPosition[0], localPosition[1], localPosition[2])
    pc = parentConstraint(trans, tempGrp, mo=0)
    delete(pc)
    position = tempLoc.getTranslation("world")
    delete(tempGrp)
    return position


def create_block_model_cmd():
    '''Creates block models for the selected joint(s).'''
    sel = ls(sl=True)
    if not sel:
        raise Exception('Select a joint to continue.')

    #check for node type joint
    joints = []
    for s in sel:
        if not isinstance(s, nodetypes.Joint):
            print 'Selection is not node type joint.  Skipping: {0}'.format(s.name())
        else:
            joints.append(s)

    for jnt in joints:
        create_block_model(jnt)


def create_block_model(jnt, parent_option=True, girth_size=3, ratio_option=True, ratio_value=0.25, skip_end_joints=True, default_dist=1):
    '''
```

What's the Problem?

- Developers can work in an IDE, but scripts can only be run and tested within Maya
  - Developers tend to work within Maya for efficiency
- Scripts can cause Maya to crash
  - Possible loss of script
- Inconvenient and time-consuming to save out versions of scripts and keep track of changes

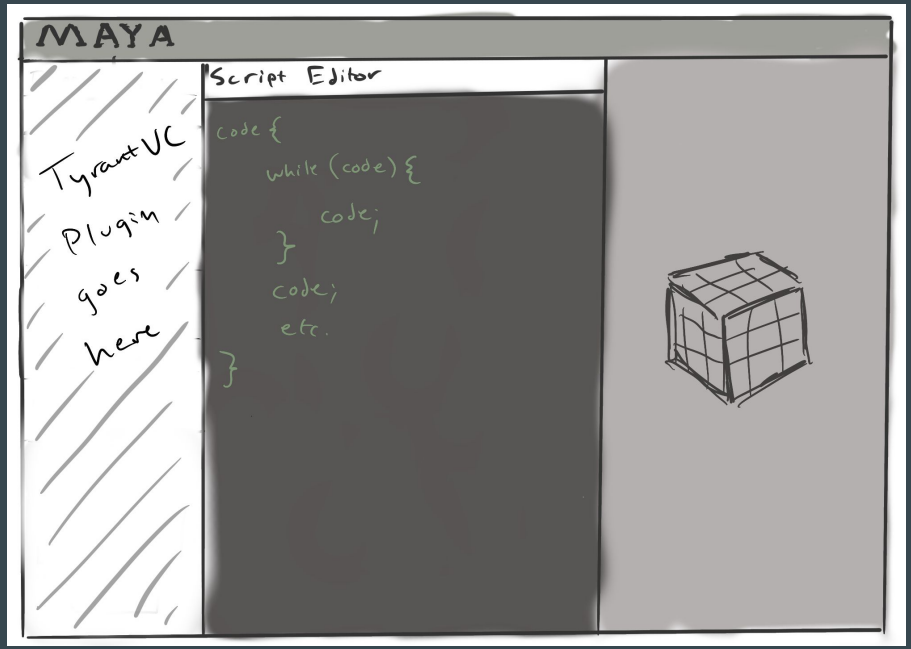
Current Solutions

- Charcoal:
  - Replaces/revamps script editor
  - Provides functionality for auto-save on script execution
  - Doesn’t offer a version control system
- GitHub Desktop
  - Version control system with an easy UI
  - Requires switching out of Maya to manage script versions and push scripts
    - Research shows that developers prefer not to do this

Our Solution: a Version Control Plugin

A plugin that would allow developers to keep track of different versions of script files and easily save a script file before running it.

This will allow for an easier time writing scripts through solving inconveniences mentioned before.



Approach

What will the plugin do?

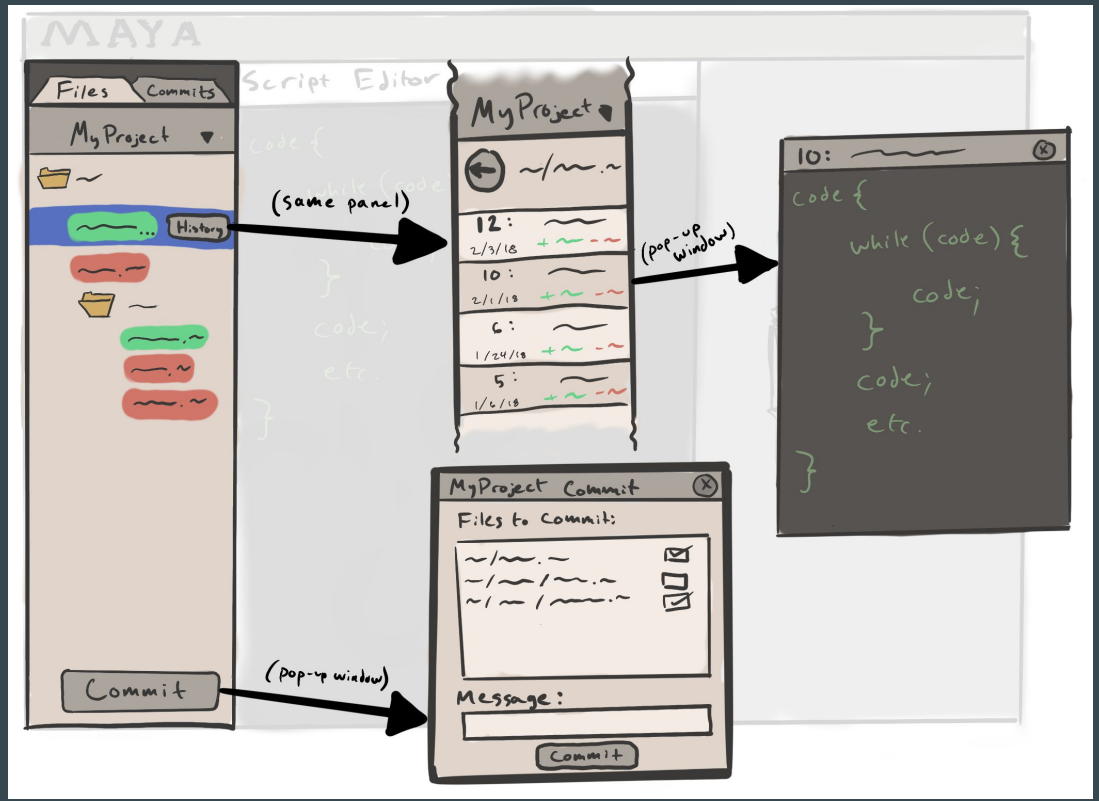
- Allows developers to see which repository they are working in
- Displays which files have been modified
- Allows users to view past committed versions of a script
- Allows for standard Git functionality of committing and pushing

Other Requirements:

- Our target audience is tech-oriented artists so our metrics are focused on usability and clarity
    - Our software must be intuitive enough for someone who doesn't use git to figure out

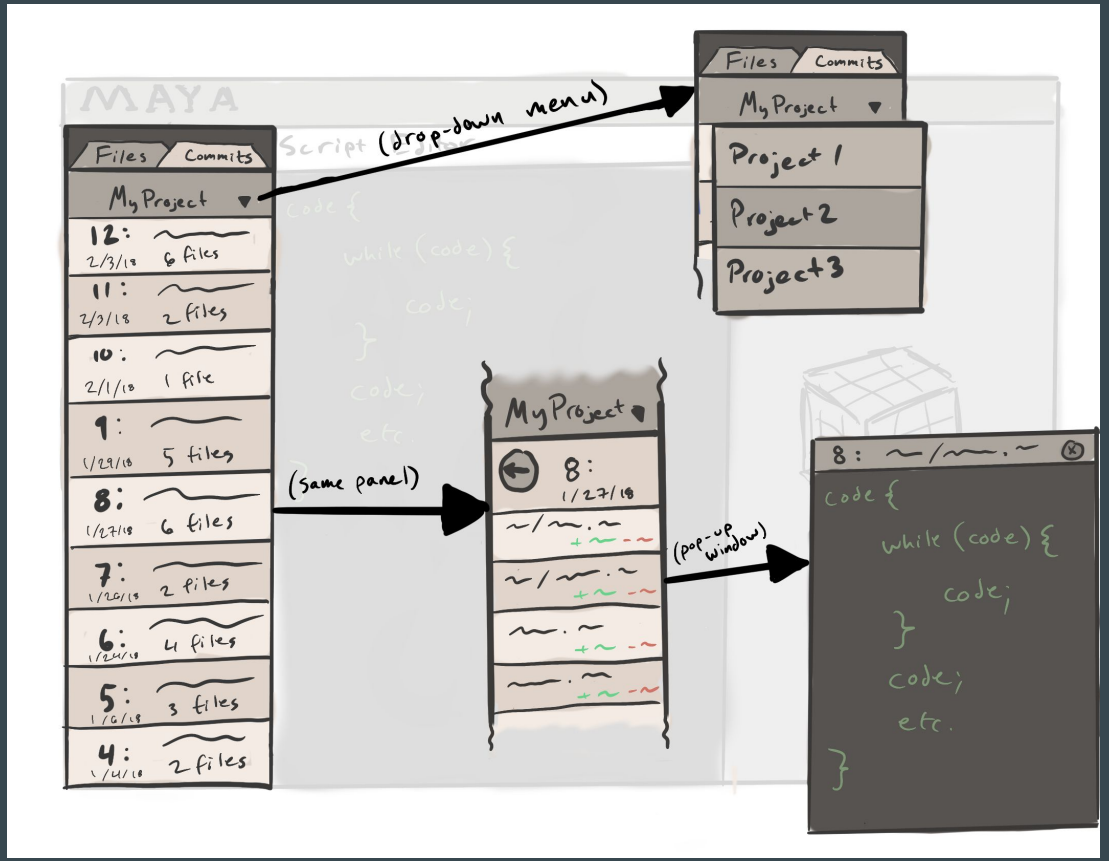
Features

- File browser
- File version history
- Script viewer (popup)
- Commit files (popup)

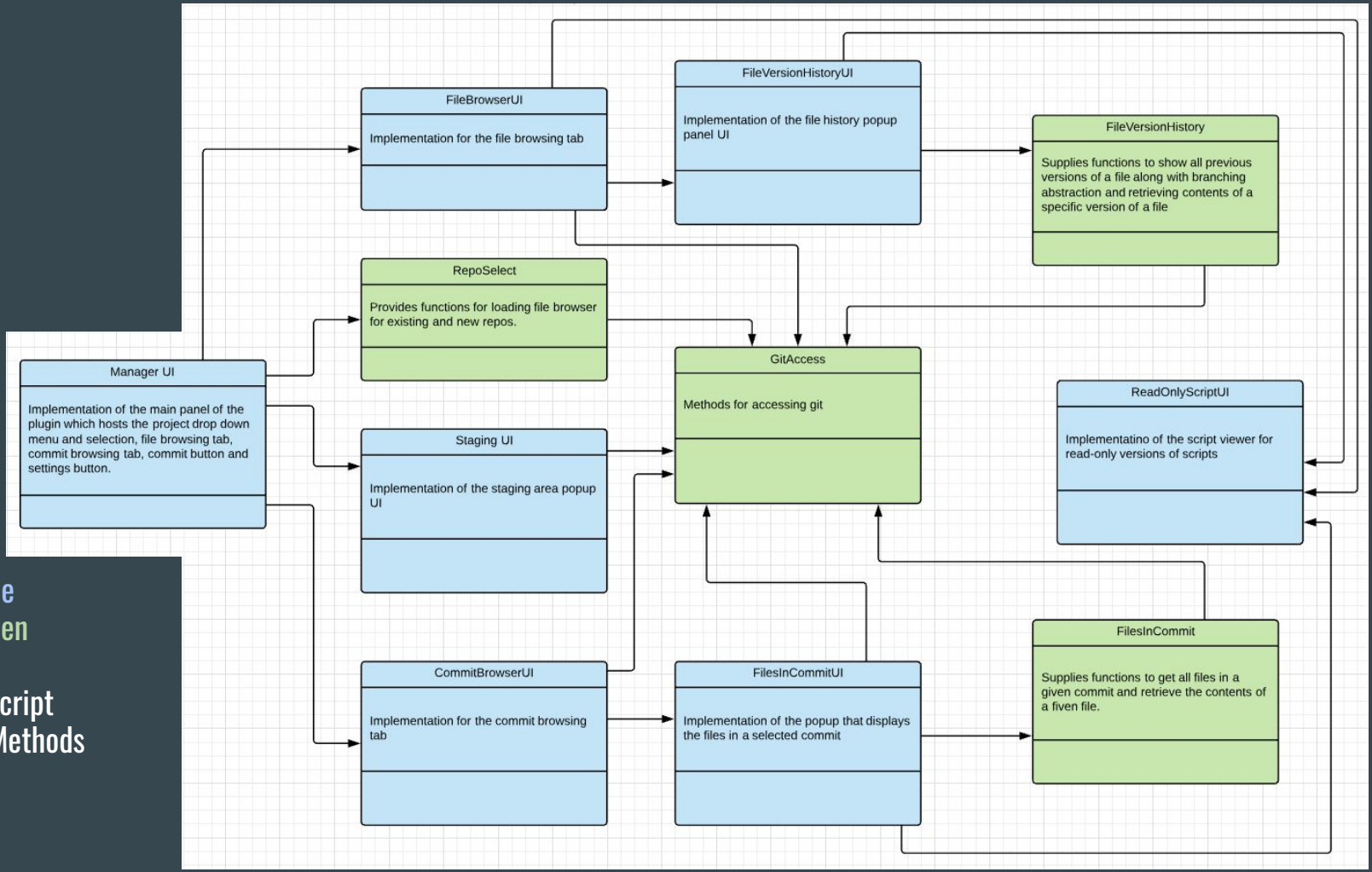


Features

- Change projects (repos)
- Commit browser
- View which files were in a commit
- View a script file from a given commit



Our Architecture



Front End = Blue
Back End = Green

Box = Python Script
Arrow = Calls Methods

Preliminary Results

- Which features to keep, which to abstract away or simply remove.
  - Branching is complicated and will be hidden away.
  - Adding, committing and pushing all combined into one action.
- Avoiding distracting users from features with ambiguous and overly complicated UI.
  - Popup windows to force users to perform an action, for example the staging area.

Our Concerns

- How intuitive our plugin is for somebody unfamiliar with git
  - Target audience is tech oriented artists who do not have much experience in software development.
- Evaluation of usability and clarity will be performed by UW Animation Lab
  - What might be clear and useful for one group might not be as clear or useful to others.

Conclusion

- Artists get into scripting to automate behaviors in larger projects.
- Have to script in Maya, yet the script editor has minimal features.
- Simplified Version Control useful for working on anything from small scripts to larger Maya focused code bases.
- UI and Architecture designs established, implementation is next.
- Evaluation will be done through user studies performed at the UW Animation Lab.