Python for Maya

Lecturer: Jan-Philipp Koch

Day 1: 16.12.20

Day 2: 13.01.21

Day 3: 25.01.21

Day 4: 03.02.21

Content

- Introduction
  - Python for Maya Course Intro.
  - Python Intro.
  - Python in Games + VFX

Content

- Introduction
  - Python for Maya Course Intro.
  - Python Intro.
  - Python in Games + VFX
- Programming in Maya

Content

- Introduction
  - Python for Maya Course Intro.
  - Python Intro.
  - Python in Games + VFX
- Programming in Maya
- Python for Maya: Basics

Content

- Introduction
  - Python for Maya Course Intro.
  - Python Intro.
  - Python in Games + VFX
- Programming in Maya
- Python for Maya: Basics
- How I used Python in the past

Content

- Introduction
  - Python for Maya Course Intro.
  - Python Intro.
  - Python in Games + VFX
- Programming in Maya
- Python for Maya: Basics
- How I used Python in the past
- How to use Python for your projects

Content

- Introduction
  - Python for Maya Course Intro.
  - Python Intro.
  - Python in Games + VFX
- Programming in Maya
- Python for Maya: Basics
- How I used Python in the past
- Task: Find a way to use Python for your projects
- Resources (useful links, books, videos)

Content

- Introduction
  - Python for Maya Course Intro.
  - Python Intro.
  - Python in Games + VFX
- Programming in Maya
- Python for Maya: Basics
- How I used Python in the past
- How to use Python for your projects
- Resources (useful links, books, videos)
- Python for Maya: Advanced

Introduction

- 2014: Rigger / Character Animator at SaveGame Studios
- 2016: B.Sc in Computer Science (Focus in Game Development)
- 2017 - Present: Pipeline Technical Director / Technical Artist at Lavalabs
  - BoxaGrippal, TVC: https://vimeo.com/240124795
  - Mein Lotta-Leben, 2019 Feature Film: https://www.youtube.com/watch?v=1ml5qpruukI
  - 7500, 2019 Feature Film: https://www.youtube.com/watch?v=B4Dbpt1c5r0&feature=youtu.be
  - Die Heinzels, 2020 Animation Feature: https://www.youtube.com/watch?v=OO8Tgdf3y_c
  - Der Überläufer, 2020 Feature: https://www.youtube.com/watch?v=FGVxPrhXqL0&ab_channel=KinoCheckHome
  - Blood & Beer, 2020 Series: Netflix
  - Yakari, 2020 Animation: https://www.youtube.com/watch?v=CmSHIDbxanM&ab_channel=KinoCheckFamilie
  - …
- 2020 - Present: Self-employed / part-time Software Engineer

https://www.linkedin.com/in/janphilippkoch/

Introduction

- Who are you?
- What is your passion?
- Programing skills?
- Prefered Software?

Python for Maya Course

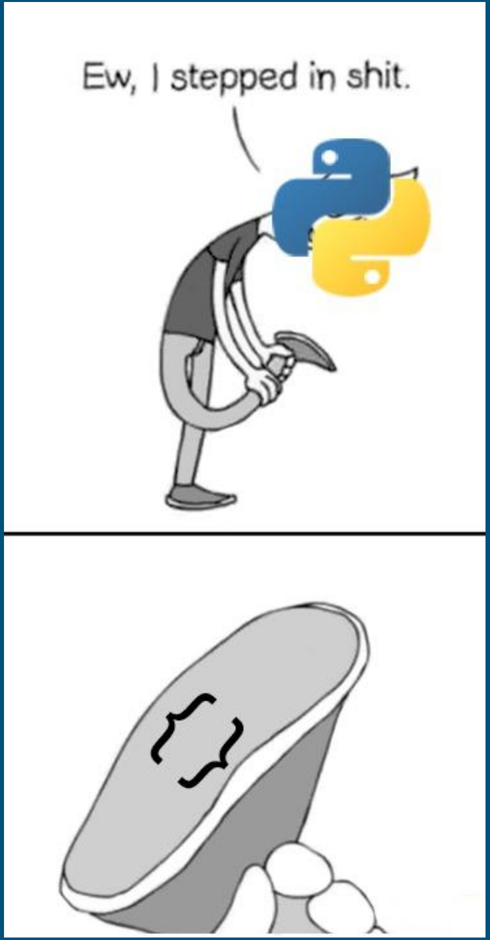
- Day 1: 16.12.20, 17:30 - 20:30 CET:
  - Introduction + Basics + Find a way to use Python for your project
- Day 2: 13.01.21, 17:30 - 20:30 CET:
  - Presentation of your ideas + Discussion elaborate on them together + work on your solutions
- Day 3: 25.01.21, 17:30 - 20:30 CET:
  - Advanced Optional Topics
- Day 4: 03.02.21, 17:30 - 20:30 CET:
  - Advanced Optional Topics

Python Intro

- Released + developed by Guido van Rossum 1991
- Easy read + learn
- Object oriented
- Python Standard library (https://docs.python.org/3.7/library/)
- Used in many industries

Why you should learn Python

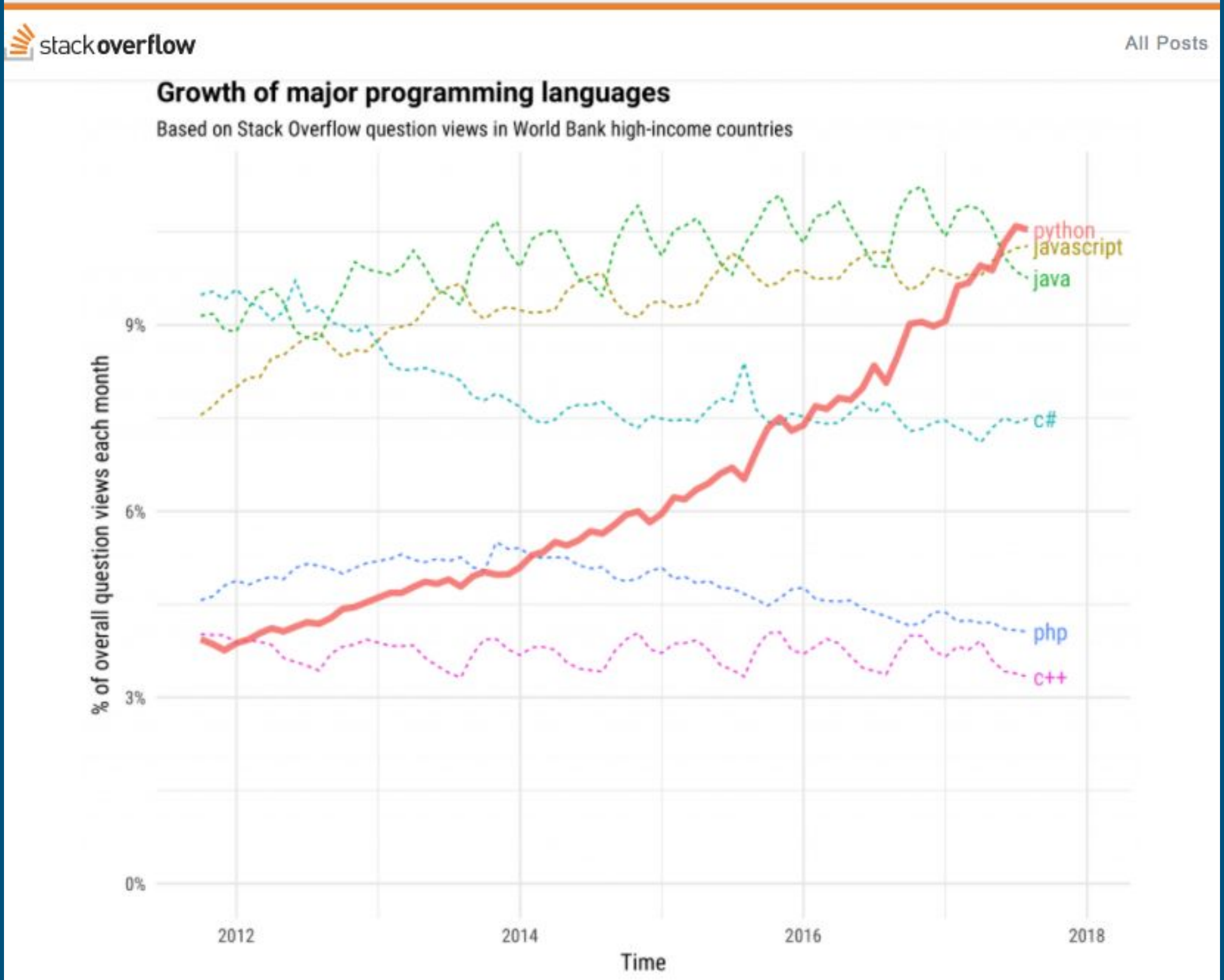
- Easy. (That’s it. Just easy. Isn't this enough?)
  - To write + read!



Why you should learn Python

- Easy. (That’s it. Just easy. Isn't this enough?)
- One of the most used programming languages.

Python Introduction



Why you should learn Python

- Easy. (That’s it. Just easy. Isn't this enough?)
- One of the most used programming languages.
- Automate the boring stuff to concentrate on creative tasks.

Why you should learn Python

- Easy. (That’s it. Just easy. Isn't this enough?)
- One of the most used programming languages.
- Automate the boring stuff to concentrate on creative tasks.
- Reduce errors on repetitive tasks.

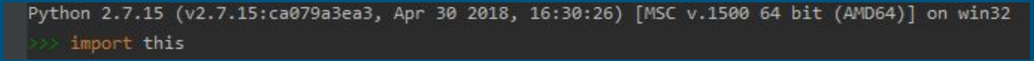
Why you should learn Python

- Easy. (That’s it. Just easy. Isn't this enough?)
- One of the most used programming languages.
- Automate the boring stuff to concentrate on creative tasks.
- Reduce errors on repetitive tasks.
- Be able to have more creative interations.

Why you should learn Python

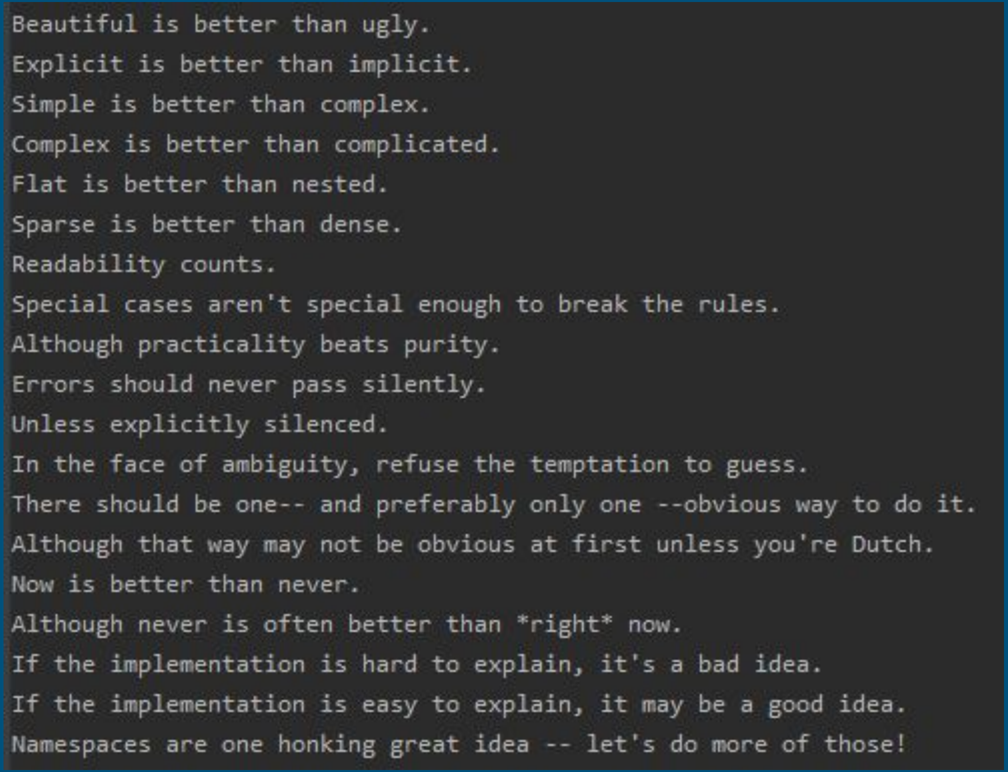
- Easy. (That’s it. Just easy. Isn't this enough?)
- One of the most used programming languages.
- Automate the boring stuff to concentrate on creative tasks.
- Reduce errors on repetitive tasks.
- Be able to have more creative interations.
- Increase chances for getting hired / get a raise.

“How to” Python

- The Zen of Python
  - 

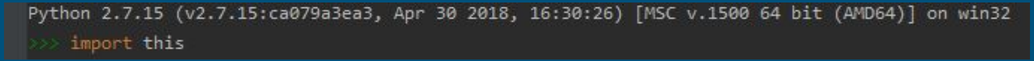
```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC v.1500 64 bit (AMD64)] on win32
>>> import this
```

Python Introduction



```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

“How to” Python

- The Zen of Python
    - 

```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC v.1500 64 bit (AMD64)] on win32
>>> import this
```

- Python standard library (https://docs.python.org/3.7/library/)
    - Ships with the python installation
    - Always try to use standard tools (more important if you plan to provide your tools for others)

Python in Games + VFX

Python in Games

- Mobile / browser

Python in Games

- Mobile / browser
- Game logic / Gameplay with Python
  - Possible but rarely the best option. But why?
    - Slow performance

Python in Games

- Mobile / browser
- Game logic / Gameplay with Python
  - Possible but rarely the best option. But why?
    - Slow performance
- Separate functionalities like chats, inventory systems, ect.

Python in Games

- Mobile / browser
- Game logic / Gameplay with Python
    - Possible but rarely the best option. But why?
        - Slow performance
- Separate functionalities like chats, inventory systems, ect.
- Used for prototyping

Python in Games

- Mobile / browser
- Game logic / Gameplay with Python
  - Possible but rarely the best option. But why?
    - Slow performance
- Separate functionalities like chats, inventory systems, ect.
- Used for prototyping
- Useful Frameworks:
  - https://www.renpy.org/
  - https://www.pygame.org/news
  - https://kivy.org/#home

Python in VFX

- Nearly every industry Software supports python. (maya, max, houdini, cinema4d, unreal, unity, nuke, heiro, ..)
    - https://vfxplatform.com/

Python in VFX

- Nearly every industry Software supports python. (maya, max, houdini, cinema4d, unreal, unity, nuke, heiro, ..)
    - https://vfxplatform.com/
- Quickly create huge improvements for workflows + software

Python in VFX

- Nearly every industry Software supports python. (maya, max, houdini, cinema4d, unreal, unity, nuke, heiro, ..)
    - https://vfxplatform.com/
- Quickly create huge improvements for workflows + software
- Finds use in nearly all occupations

Programming in Maya

- ???

Programming in Maya

- MEL
  - MEL can be quicker to “hack” something down but try to avoid it
  - Not all MEL commands are supported in python commands!

Programming in Maya

- MEL
  - MEL can be quicker to “hack” something down but try to avoid it
  - Not all MEL commands are supported in python commands!
- C++
  - Maya is build with C++
  - Best performance
  - Not many realistic use-cases in production

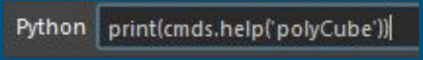
Programming in Maya

- MEL
  - MEL can be quicker to “hack” something down but try to avoid it
  - Not all MEL commands are supported in python commands!
- C++
  - Maya is build with C++
  - Best performance
  - Not many realistic use-cases in production
- Python:
  - Python commands: maya.cmds (mel wrapper)
    - Sufficient for nearly all task around maya
  - Openmaya: maya.om (link)
    - More pythonic
    - Better performance

Basics

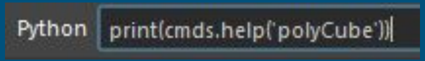
- Open Maya!

Basics

- Open Maya!
- Maya doc (link)
  - Or use 

```
print(cmds.help('polyCube'))
```

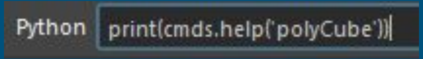
Basics

- Open Maya!
- Maya doc (link)
  - Or use 

```
print(cmds.help('polyCube'))
```

- Script Editor History (Echo all commands)

Basics

- Open Maya!
- Maya doc (link)
    - Or use 

```
print(cmds.help('polyCube'))
```

- Script Editor History (Echo all commands)
- maya.cmds “modes”
    - create, edit, query

Basics: Scatter cubes

- What you will learn:
  - How to use the basic tools for developing with python in maya
  - How to make your code more reusable

Basics: Scatter cubes

1. Create a cube
2. Create a lot of cubes
3. Cubes should be placed randomly (XYZ 1-1000)
4. Cubes should be sized randomly (1-10)
5. Bonus: ???

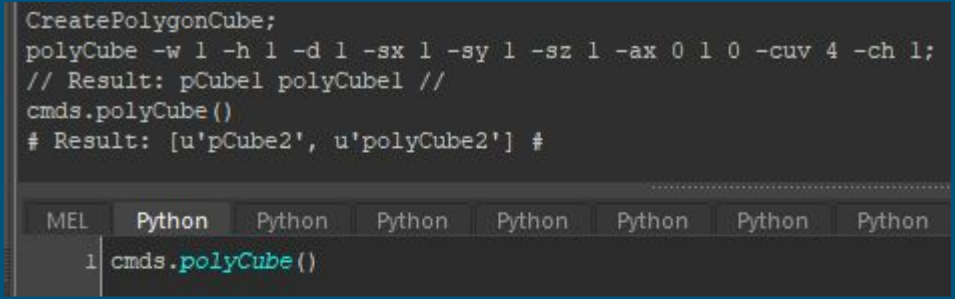
Time: ca. 20 min

www.ToneDeafComics.com

© 2016 John Bogenschutz

Basics: Scatter cubes (Breakdown)

1. Create a cube
   a. Create one by hand
   b. Check which command was used
   c. Check online/offline doc
   d. Make a cube with python

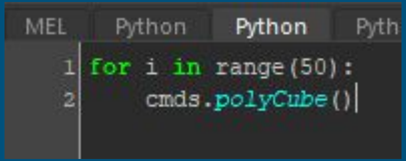


```
CreatePolygonCube;
polyCube -w 1 -h 1 -d 1 -sx 1 -sy 1 -sz 1 -ax 0 1 0 -cuv 4 -ch 1;
// Result: pCube1 polyCube1 //
cmds.polyCube()
# Result: [u'pCube2', u'polyCube2'] #
```

```
cmds.polyCube()
```

Basics: Scatter cubes (Breakdown)

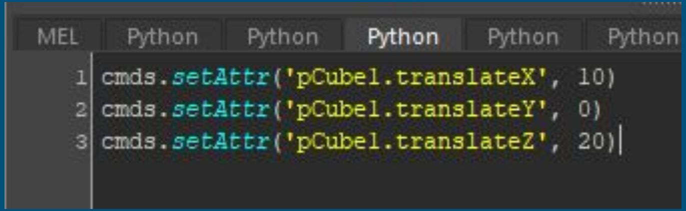
1. Create a cube
2. Create a lot of cubes
    a. For loop to create a lot of cubes with python



```
for i in range(50):
    cmds.polyCube()
```

Basics: Scatter cubes (Breakdown)

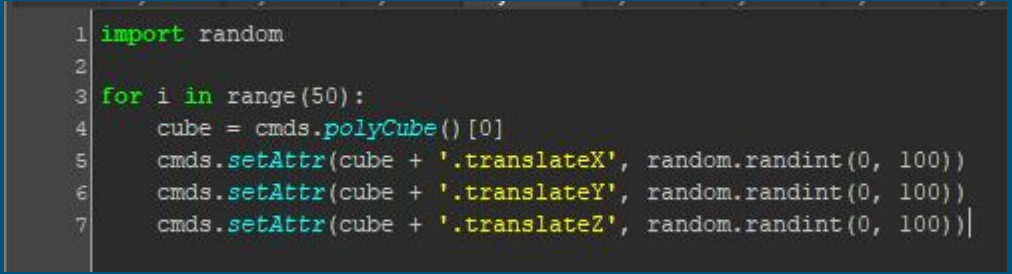
1. Create a cube
2. Create a lot of cubes
3. Cubes should be placed randomly (XYZ 1-100)
   a. Can we give the polyCube transform information on create?
   b. We need a new command! setAttr



```python
cmds.setAttr('pCube1.translateX', 10)
cmds.setAttr('pCube1.translateY', 0)
cmds.setAttr('pCube1.translateZ', 20)
```

Basics: Scatter cubes (Breakdown)

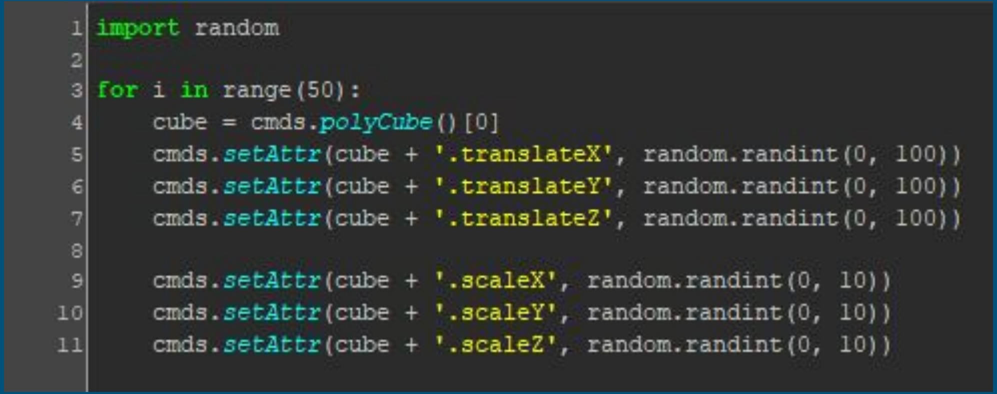
1. Create a cube
2. Create a lot of cubes
3. Cubes should be placed randomly (XYZ 1-100)
    a. Can we give the polyCube transform information on create?
    b. We need a new command! setAttr
    c. Replace numbers with random number



```
import random

for i in range(50):
    cube = cmds.polyCube()[0]
    cmds.setAttr(cube + '.translateX', random.randint(0, 100))
    cmds.setAttr(cube + '.translateY', random.randint(0, 100))
    cmds.setAttr(cube + '.translateZ', random.randint(0, 100))
```

Basics: Scatter cubes (Breakdown)

1. Create a cube
2. Create a lot of cubes
3. Cubes should be placed randomly (XYZ 1-100)
4. Cubes should be sized randomly (1-10)



```python
import random

for i in range(50):
    cube = cmds.polyCube()[0]
    cmds.setAttr(cube + '.translateX', random.randint(0, 100))
    cmds.setAttr(cube + '.translateY', random.randint(0, 100))
    cmds.setAttr(cube + '.translateZ', random.randint(0, 100))

    cmds.setAttr(cube + '.scaleX', random.randint(0, 10))
    cmds.setAttr(cube + '.scaleY', random.randint(0, 10))
    cmds.setAttr(cube + '.scaleZ', random.randint(0, 10))
```

Basics: Scatter cubes (Breakdown)

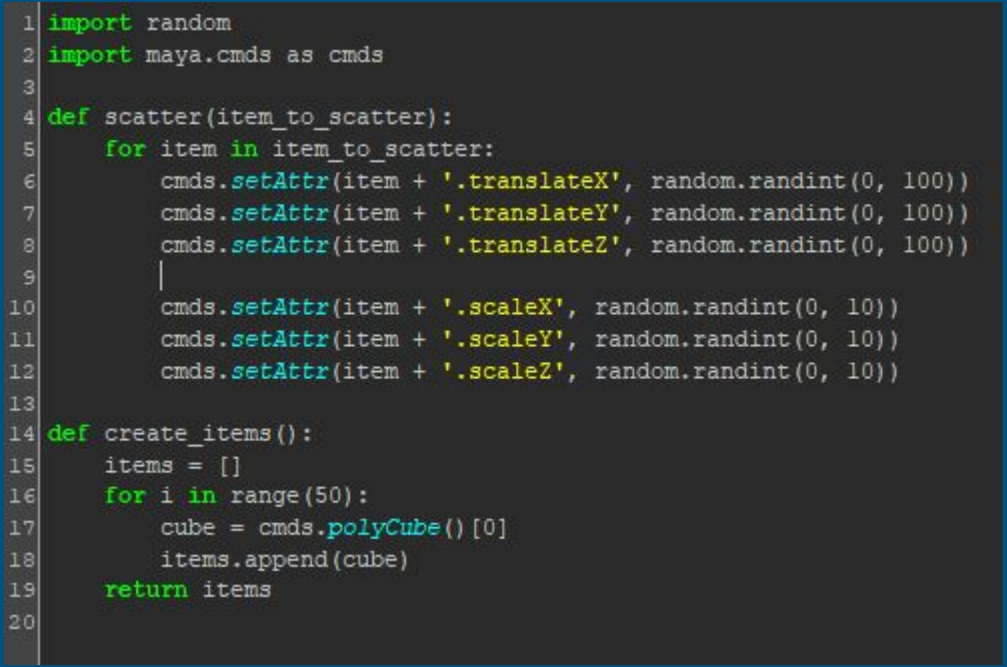
1. Create a cube
2. Create a lot of cubes
3. Cubes should be placed randomly (XYZ 1-100)
4. Cubes should be sized randomly (1-10)
5. Bonus: ???

Basics

- Reusability
    - What are ways to make your cube scatter more useful + reusable?

Basics

- Reusability



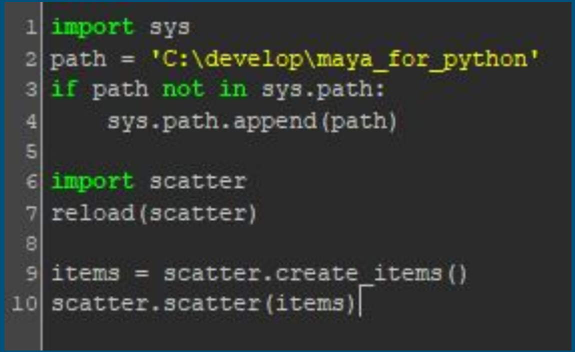
```python
import random
import maya.cmds as cmds

def scatter(item_to_scatter):
    for item in item_to_scatter:
        cmds.setAttr(item + '.translateX', random.randint(0, 100))
        cmds.setAttr(item + '.translateY', random.randint(0, 100))
        cmds.setAttr(item + '.translateZ', random.randint(0, 100))

        cmds.setAttr(item + '.scaleX', random.randint(0, 10))
        cmds.setAttr(item + '.scaleY', random.randint(0, 10))
        cmds.setAttr(item + '.scaleZ', random.randint(0, 10))

def create_items():
    items = []
    for i in range(50):
        cube = cmds.polyCube()[0]
        items.append(cube)
    return items

```

Basics

- Reusability
  - Save your script + import it into maya

Basics

- Reusability
    - Save your script + import it into maya



```
import sys
path = 'C:\develop\maya_for_python'
if path not in sys.path:
    sys.path.append(path)

import scatter
reload(scatter)

items = scatter.create_items()
scatter.scatter(items)
```

Basics

- Reusability
  - What are ways to make your cube scatter more useful + reusable?
  - Always aim for good readability for your variables, function + class names and parameters.

Basics

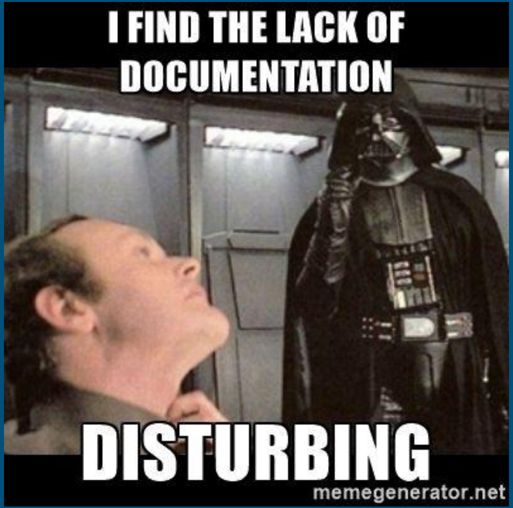
- Reusability
  - What are ways to make your cube scatter more useful + reusable?
  - Always aim for good readability for your variables, function + class names and parameters.
  - DRY - Principle (don’t repeat yourself)

Basics

- Reusability
  - What are ways to make your cube scatter more useful + reusable?
  - Always aim for good readability for your variables, function + class names and parameters.
  - DRY - Principle (don’t repeat yourself)
  - Try to focus on it early on will save you time afterwards.

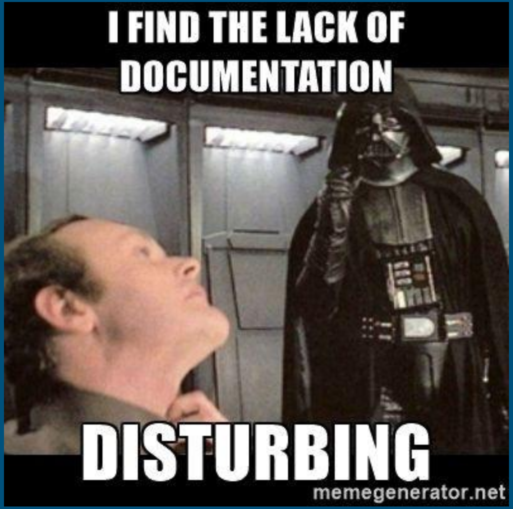
Basics

- Documentation



Basics

- Documentation
  - # - line comments
  - “”” triple quotes “”” - docstrings
  - help(modulename)



Maya’s userSetup.py

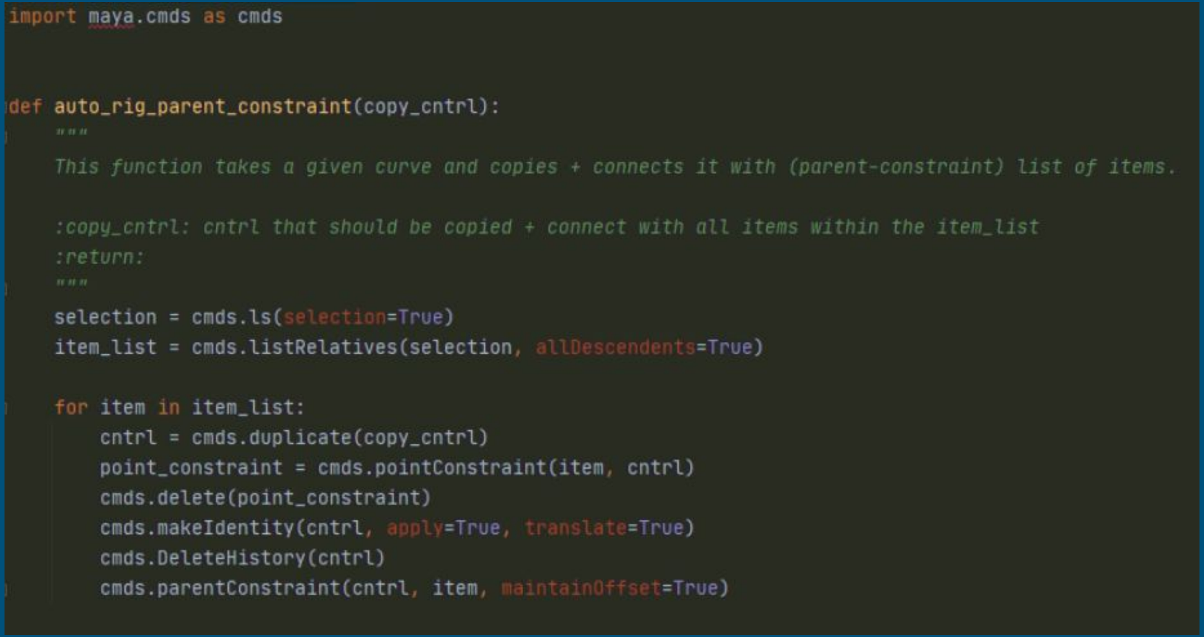
- userSetup.py vs. userSetup.mel (.mel gets loaded before the .py)
- Will be runned each time maya starts
- Great way to add to the sys.path of maya’s python (to add your custom tools)
- Set env variables, enable plugins, ect.
- Location:
  - Windows:
    - C:\Users\YOURUSER\Documents\maya\scripts
    - C:\Users\YOURUSER\Documents\maya\<version>\scripts
  - OS X:
    - ~Library/Preferences/Autodesk/maya/<version>/scripts/
  - Linux:
    - ~/maya/<version>/scripts/

How I used Python in the past:

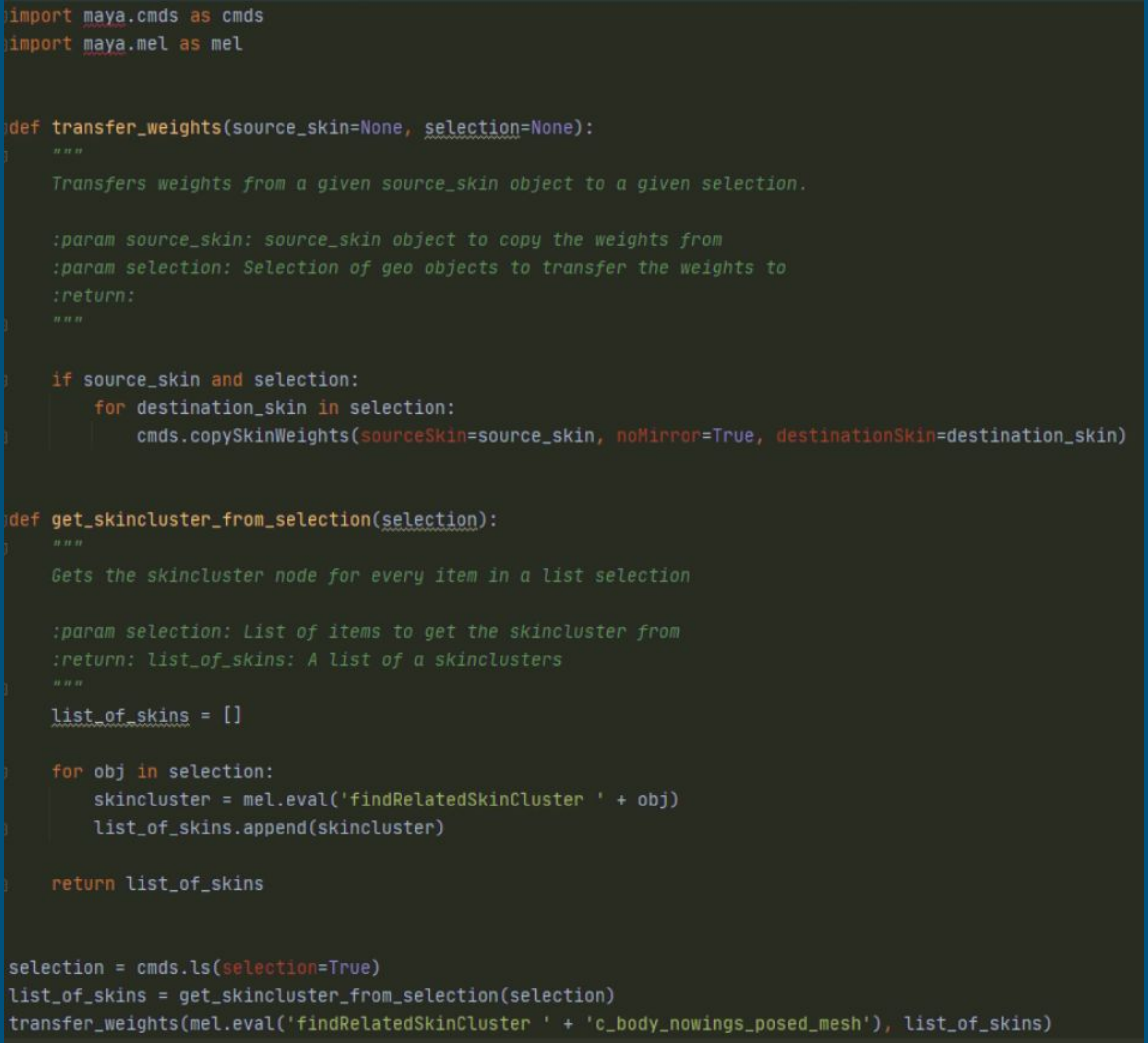
- Rigging: Autorigger, Picker, ...

How I used Python in the past:

- Rigging: Autorigger, Picker, ...



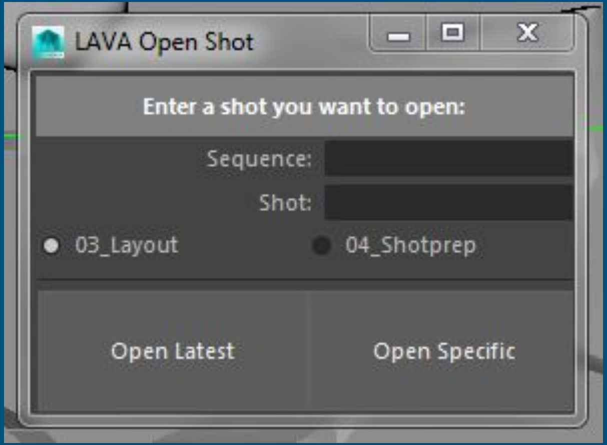
```
import maya.cmds as cmds


def auto_rig_parent_constraint(copy_cntrl):
    """
    This function takes a given curve and copies + connects it with (parent-constraint) list of items.

    :copy_cntrl: cntrl that should be copied + connect with all items within the item_list
    :return:
    """
    selection = cmds.ls(selection=True)
    item_list = cmds.listRelatives(selection, allDescendents=True)

    for item in item_list:
        cntrl = cmds.duplicate(copy_cntrl)
        point_constraint = cmds.pointConstraint(item, cntrl)
        cmds.delete(point_constraint)
        cmds.makeIdentity(cntrl, apply=True, translate=True)
        cmds.DeleteHistory(cntrl)
        cmds.parentConstraint(cntrl, item, maintainOffset=True)
```



```python
import maya.cmds as cmds
import maya.mel as mel


def transfer_weights(source_skin=None, selection=None):
    """
    Transfers weights from a given source_skin object to a given selection.

    :param source_skin: source_skin object to copy the weights from
    :param selection: Selection of geo objects to transfer the weights to
    :return:
    """

    if source_skin and selection:
        for destination_skin in selection:
            cmds.copySkinWeights(sourceSkin=source_skin, noMirror=True, destinationSkin=destination_skin)


def get_skincluster_from_selection(selection):
    """
    Gets the skincluster node for every item in a list selection

    :param selection: List of items to get the skincluster from
    :return: list_of_skins: A list of a skinclusters
    """
    list_of_skins = []

    for obj in selection:
        skincluster = mel.eval('findRelatedSkinCluster ' + obj)
        list_of_skins.append(skincluster)

    return list_of_skins


selection = cmds.ls(selection=True)
list_of_skins = get_skincluster_from_selection(selection)
transfer_weights(mel.eval('findRelatedSkinCluster ' + 'c_body_nowings_posed_mesh'), list_of_skins)
```

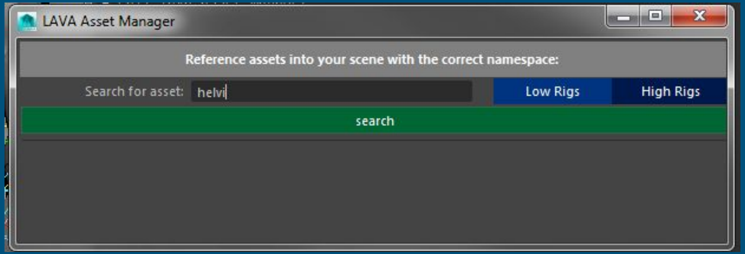
How I used Python in the past:

- Rigging: Autorigger, Picker, ...
- Automation: Scene setup, reference handling, data reports, error reports, assetmanager



LAVA Open Shot

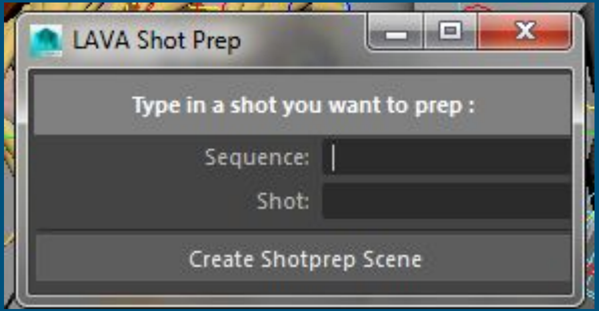
Enter a shot you want to open:



LAVA Asset Manager

Reference assets into your scene with the correct namespace:

Search for asset: helvi



LAVA Shot Prep

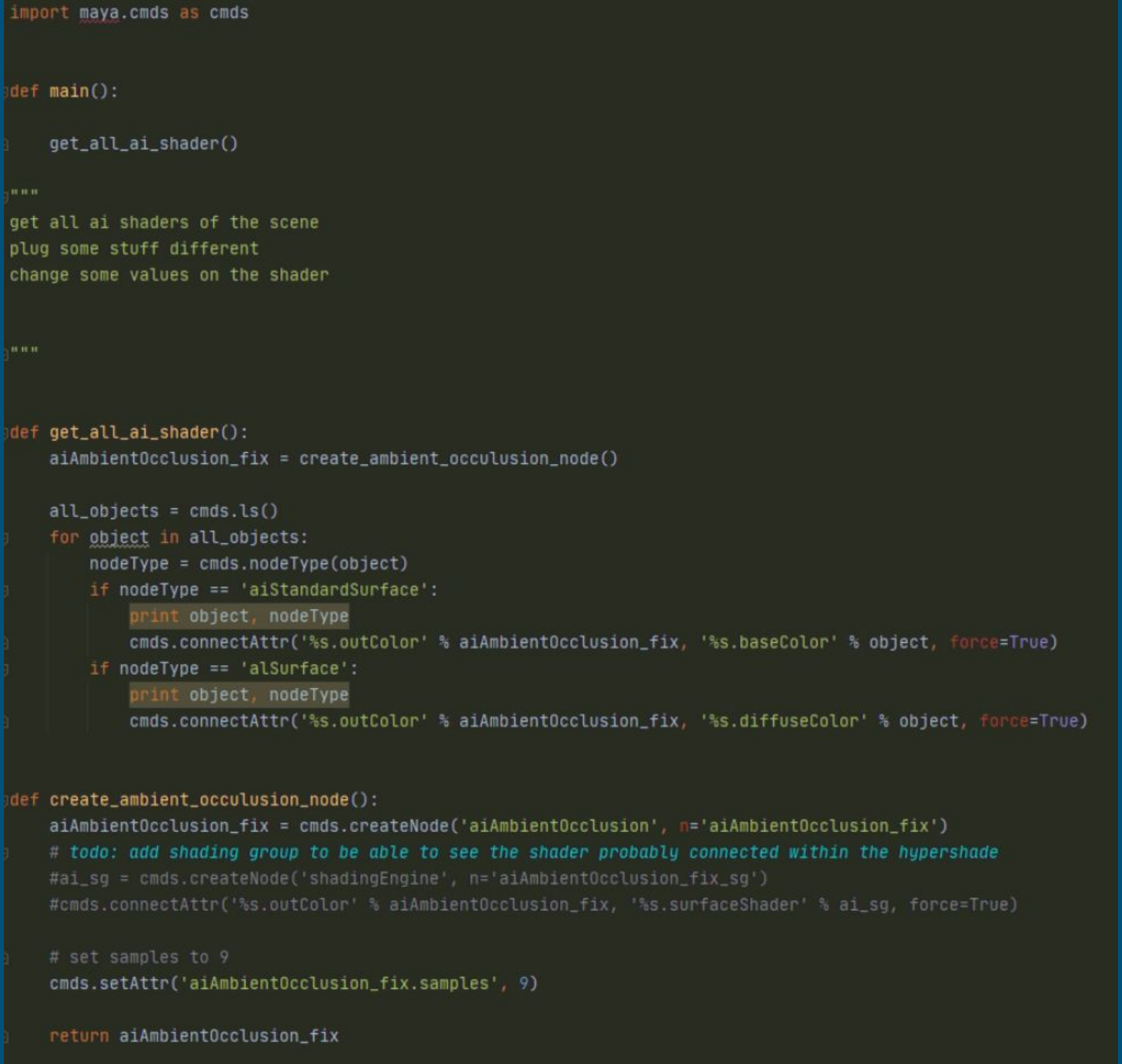
Type in a shot you want to prep :

How I used Python in the past:

- Rigging: Autorigger, Picker
- Automation: Scene setup, reference handling, data reports, error reports, assetmanager
- Heinzels 2020 (link)

How I used Python in the past:

- Rigging: Autorigger, Picker
- Automation: Scene setup, reference handling, data reports, error reports, assetmanager
- Heinzels 2020 (link)
- Editing multiple Maya Nodes/Values/Settings at once



```python
import maya.cmds as cmds


def main():

    get_all_ai_shader()

"""
get all ai shaders of the scene
plug some stuff different
change some values on the shader


"""


def get_all_ai_shader():
    aiAmbientOcclusion_fix = create_ambient_occulusion_node()

    all_objects = cmds.ls()
    for object in all_objects:
        nodeType = cmds.nodeType(object)
        if nodeType == 'aiStandardSurface':
            print object, nodeType
            cmds.connectAttr('%s.outColor' % aiAmbientOcclusion_fix, '%s.baseColor' % object, force=True)
        if nodeType == 'alSurface':
            print object, nodeType
            cmds.connectAttr('%s.outColor' % aiAmbientOcclusion_fix, '%s.diffuseColor' % object, force=True)


def create_ambient_occulusion_node():
    aiAmbientOcclusion_fix = cmds.createNode('aiAmbientOcclusion', n='aiAmbientOcclusion_fix')
    # todo: add shading group to be able to see the shader probably connected within the hypershade
    #ai_sg = cmds.createNode('shadingEngine', n='aiAmbientOcclusion_fix_sg')
    #cmds.connectAttr('%s.outColor' % aiAmbientOcclusion_fix, '%s.surfaceShader' % ai_sg, force=True)

    # set samples to 9
    cmds.setAttr('aiAmbientOcclusion_fix.samples', 9)

    return aiAmbientOcclusion_fix
```

How I used Python in the past:

- Rigging: Autorigger, Picker
- Automation: Scene setup, reference handling, data reports, error reports, assetmanager
- Heinzels 2020 (link)
- Editing multiple Maya Nodes/Values/Settings at once
- Preparing scenes for publish (sometimes you need different settings for working vs. publishing)

How did you use Python in the past?

Task: Find a way to use Python for your project

- Ask yourself these questions:
  - Will I do this again in the future?
  - If I code this does it give me more creative freedom?
  - Is this task error prone when done by humans?

Task: Find a way to use Python for your project

- Ask yourself these questions:
  - Will I do this again in the future?
  - If I code this does it give me more creative freedom?
  - Is this task error prone when done by humans?

whatever you're saying, the answer is YES!
more awesome pictures at THEMETAPICTURE.COM

Yes!

Task: Find a way to use Python for your project

- Ask yourself these questions:
  - Will I do this again in the future?
  - If I code this does it give me more creative freedom?
  - Is this task error prone when done by humans?

!!! Find a use-case for python in your project and mail it to janphkoch@gmail.com or send it via Slack until 13.01.21 !!!

Note: It does not need to be huge or complicated. If it solves a problem and or answers one of the above questions it will be sufficient. On the 13.01.21 everyone will present their idea and we can discuss and elaborate together on them.

Examples

https://github.com/JanPhKoch/JPK-PythonForMaya

Resources

- Books:
  - Maya Python for Games and Film
  - Python Tricks the Book (to dive deeper into cool features of python)
- Youtube:
  - Corey Schafer (link)
  - Real Python (link)
- Podcast:
  - Talk Python to me (link)
- Mailing List: Python Programming for Autodesk Maya (link)

Python for Maya

Lecturer: Jan-Philipp Koch

Day 1: 16.12.20

Day 2: 13.01.21

Day 3: 25.01.21

Day 4: 03.02.21

Python for Maya Course

- Day 1: 16.12.20, 17:30 - 20:30 CET:
  - Introduction + Basics + Find a way to use Python for your project
- **Day 2: 13.01.21, 17:30 - 20:30 CET:**
  - **Presentation of your ideas + Discussion elaborate on them together + work on your solutions**
- Day 3: 25.01.21, 17:30 - 20:30 CET:
  - Work on your solutions + Advanced Optional Topics
- Day 4: 03.02.21, 17:30 - 20:30 CET:
  - Advanced Optional Topics

Python for Maya

Lecturer: Jan-Philipp Koch

Day 1: 16.12.20

Day 2: 13.01.21

Day 3: 25.01.21

Day 4: 03.02.21

Task: Maya Asset Duplicator

You are the only layout artist on a 3d animated feature production.

Your maya scene is currently empty but you will be provided with rigs(references). Since the references are located under transform nodes that are required by the studio pipeline you can not use mayas duplicate special.

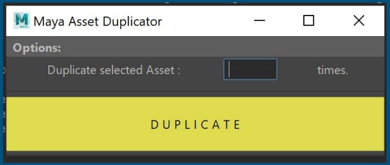
You know that you probably need to duplicate a lot of assets, for populating your scene. You want to be able to type into a GUI the amount of duplicates you want to create and based on your selection the proper node should be created.

(You should design your tool to handle different asset-types in the future)

Task: Maya Asset Duplicator

Breakdown:

- Need to duplicate maya references
- Select existing asset in the scene.
- Identify asset-type of selection (for other asset-types in the future).
- Little GUI to enter your duplication amount.



Maya Asset Duplicator

Options:

Duplicate selected Asset : ____ times.

Note: download/copy starting file from: https://github.com/JanPhKoch/JPK-PythonForMaya/blob/master/examples/asset_duplicator_start.py

Python for Maya

Lecturer: Jan-Philipp Koch

Day 1: 16.12.20

Day 2: 13.01.21

Day 3: 25.01.21

Day 4: 03.02.21

Python for Maya Course

- Day 1: 16.12.20, 17:30 - 20:30 CET:
  - Introduction + Basics + Find a way to use Python for your project
- Day 2: 13.01.21, 17:30 - 20:30 CET:
  - Presentation of your ideas + Discussion elaborate on them together + work on your solutions
- Day 3: 25.01.21, 17:30 - 20:30 CET:
  - Coding Example + Work on your solutions + Advanced Optional Topics
- **Day 4: 03.02.21, 17:30 - 20:30 CET:**
  - **Advanced Optional Topics**

Advanced Topics

- Work as a Pipeline TD
- Creating GUI
- Error handling / exception handling
- Creating maya plugins
- Version control with GIT and Github
- Maya Standalone
- Maya Testing

Work as a Pipeline Technical Director (TD)

- Agenda:
  - Basic “Definition” of Pipeline Technical Direction
  - Tasks and responsibilities of a Pipeline TD
  - Required Technology Stack

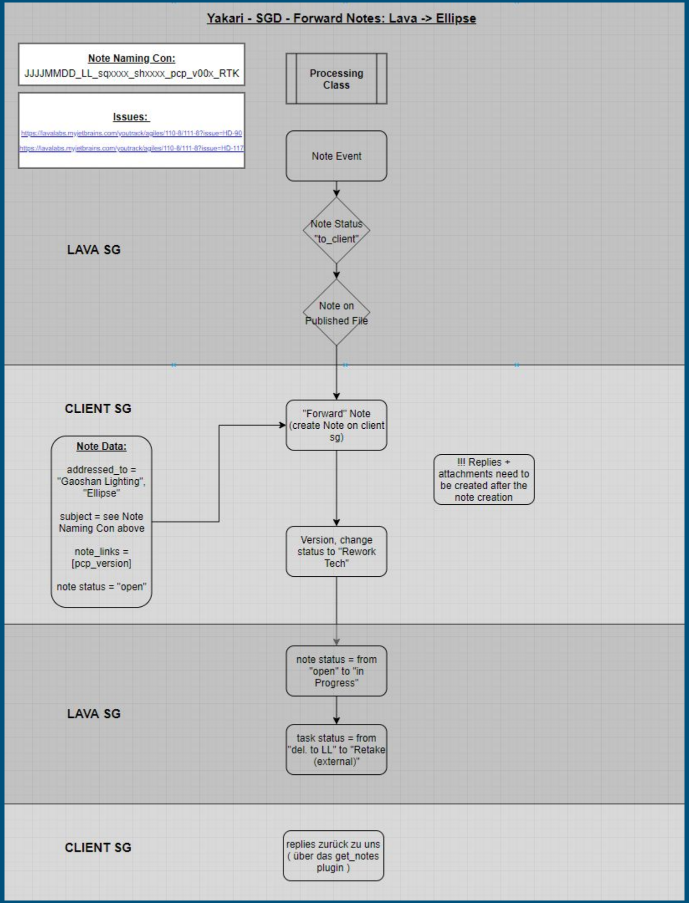
Basic “Definition” of Pipeline TD

- Design + creation + maintenance
  of the communication between separated tools and workflows
- Thrive for sophisticated “completed” solutions not quick fire fighting solutions
- Actual Job differs from studio to studio
  - small studios: ~generalized TD’s that might also work as a part time artist
  - bigger studios ~specialized TD’s with a distinct focus on pipeline development

Tasks and responsibilities of Pipeline TD’s

- **Design** + creation + maintenance
  of the communication between separated tools and workflows
  - Design:
    - Understand requirements of the clients (artists, producer, art director,...)
    - Write it down!
    - Flowcharts (or similar) tools help to visualize solutions for clients

Advanced Topics



Yakari - SGD - Forward Notes: Lava -> Ellipse

Note Naming Con:
JJJJMMDD_LL_sqxxxx_shxxxx_pcp_v00x_RTK

Issues:
https://lavalabs.myjetbrains.com/youtrack/agiles/110-8/111-8?issue=HD-90
https://lavalabs.myjetbrains.com/youtrack/agiles/110-8/111-8?issue=HD-117

Processing Class

LAVA SG

Note Event

Note Status "to_client"

Note on Published File

CLIENT SG

"Forward" Note (create Note on client sg)

Note Data:

addressed_to = "Gaoshan Lighting", "Ellipse"

subject = see Note Naming Con above

note_links = [pcp_version]

note status = "open"

Version, change status to "Rework Tech"

!!! Replies + attachments need to be created after the note creation

LAVA SG

note status = from "open" to "in Progress"

task status = from "del. to LL" to "Retake (external)"

CLIENT SG

replies zurück zu uns (über das get_notes plugin)

Tasks and responsibilities of Pipeline TD’s

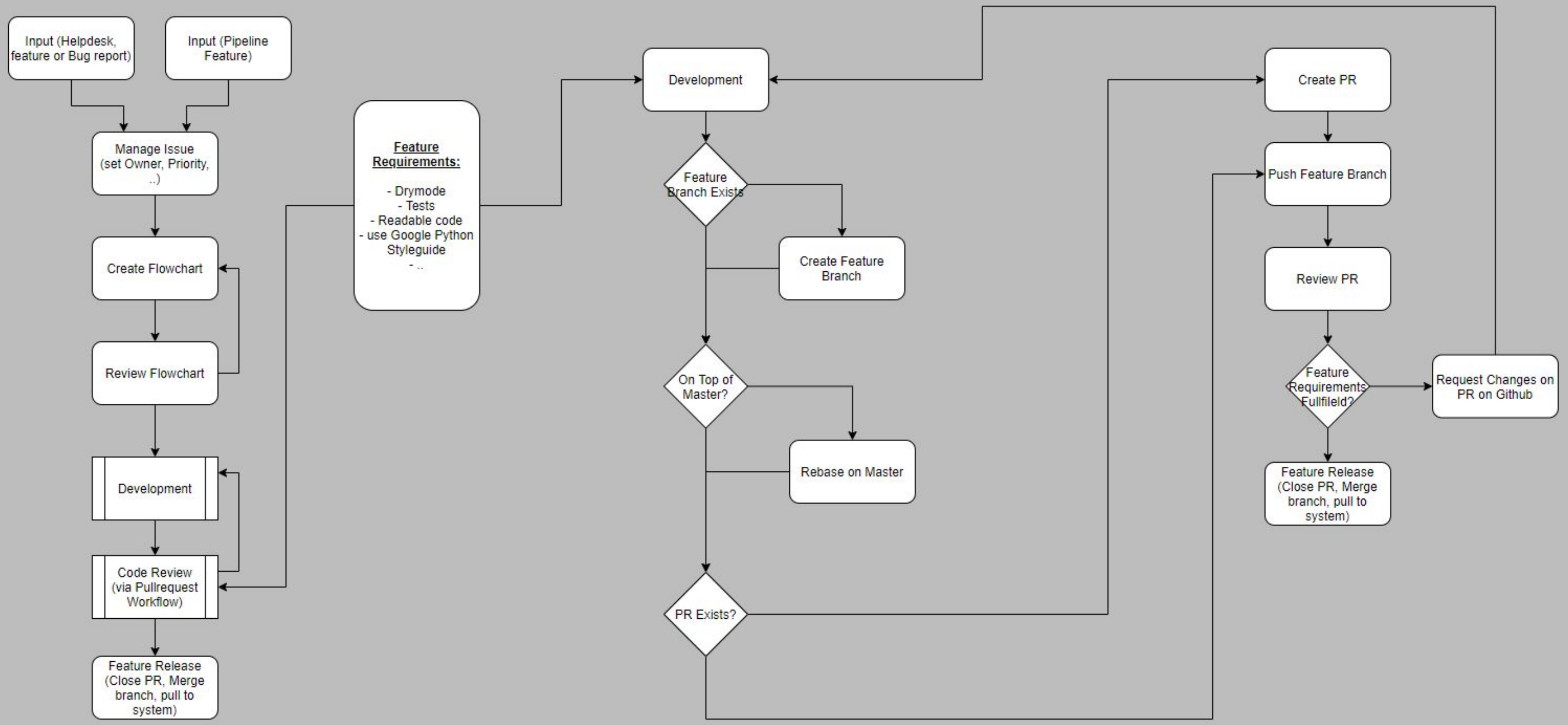
- Design + **creation** + maintenance
  of the communication between separated tools and workflows
  - Creation (Software Engineering):
    - Software Engineering Principles:
      - Separation of Concerns
      - Modularity
      - Abstraction
      - Anticipation of Change
      - Generality
      - Incremental Development
      - Consistency

https://www.d.umn.edu/~gshute/softeng/principles.html

Development Workflow

Feature Development

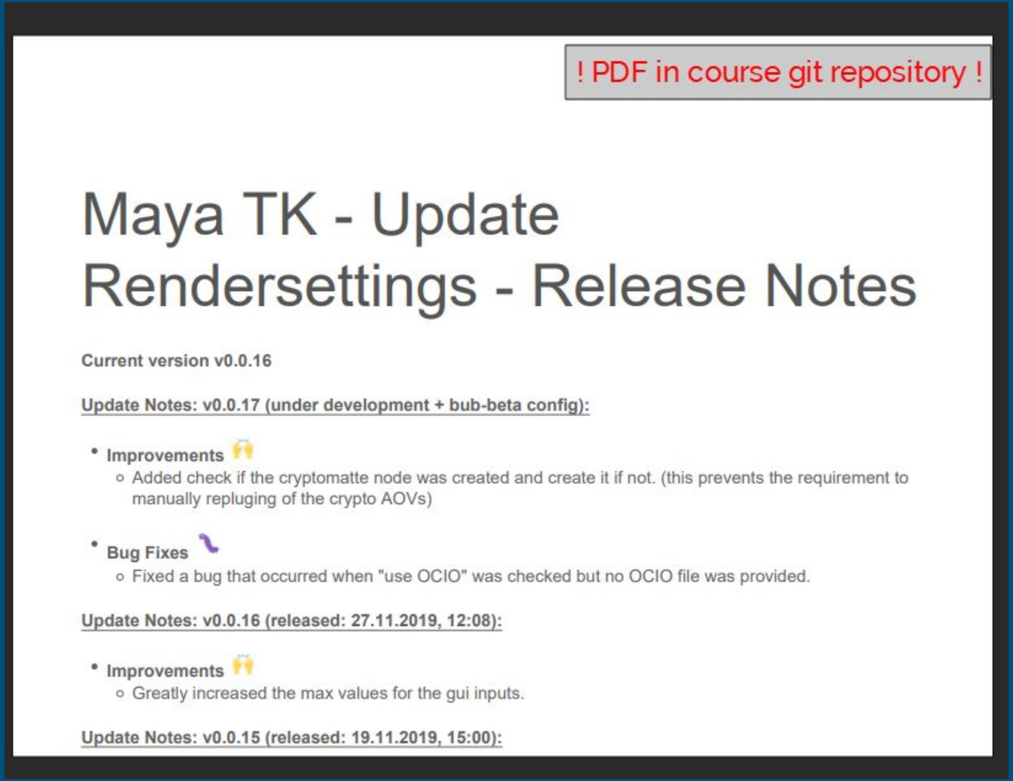
Code Review (via Pullrequest Workflow)


Input (Helpdesk, feature or Bug report)
Input (Pipeline Feature)
Manage Issue (set Owner, Priority, ..)
Create Flowchart
Review Flowchart
Development
Code Review (via Pullrequest Workflow)
Feature Release (Close PR, Merge branch, pull to system)
Feature Requirements:
- Drymode
- Tests
- Readable code
- use Google Python Styleguide
- ..
Development
Feature Branch Exists
Create Feature Branch
On Top of Master?
Rebase on Master
PR Exists?
Create PR
Push Feature Branch
Review PR
Feature Requirements Fullfileld?
Request Changes on PR on Github
Feature Release (Close PR, Merge branch, pull to system)

Tasks and responsibilities of Pipeline TD’s

- Design + creation + **maintenance**
  of the communication between separated tools and workflows
  - Maintenance:
    - Good Documentation / Communication
      - Code
      - Workflows
      - Requests and bug reports (helpdesk)
    - Reliable error handling
    - Automated systems that inform you when something went wrong

Advanced Topics



! PDF in course git repository !

Maya TK - Update Rendersettings - Release Notes

Current version v0.0.16

Update Notes: v0.0.17 (under development + bub-beta config):

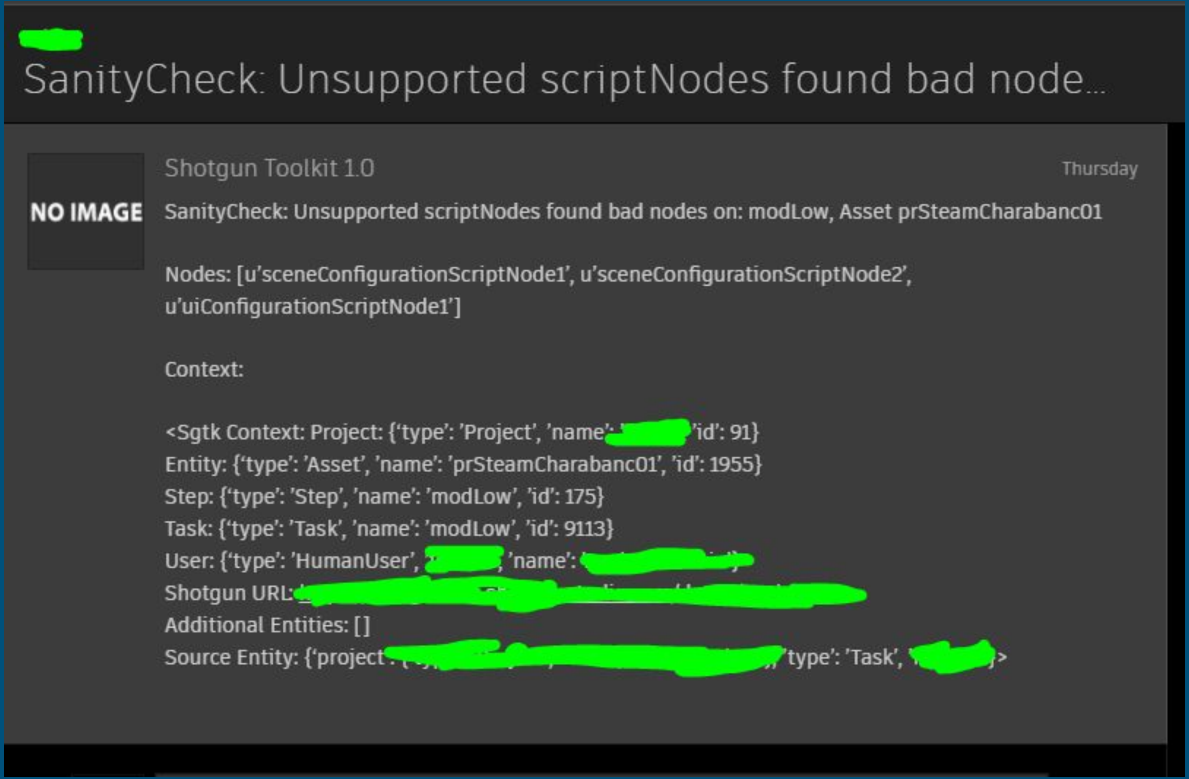
- **Improvements** 🙌
  - Added check if the cryptomatte node was created and create it if not. (this prevents the requirement to manually repluging of the crypto AOVs)
- **Bug Fixes** 🐛
  - Fixed a bug that occurred when "use OCIO" was checked but no OCIO file was provided.

Update Notes: v0.0.16 (released: 27.11.2019, 12:08):

- **Improvements** 🙌
  - Greatly increased the max values for the gui inputs.

Update Notes: v0.0.15 (released: 19.11.2019, 15:00):

Advanced Topics



[redacted]

SanityCheck: Unsupported scriptNodes found bad node...

NO IMAGE

Shotgun Toolkit 1.0

Thursday

```
SanityCheck: Unsupported scriptNodes found bad nodes on: modLow, Asset prSteamCharabanc01

Nodes: [u’sceneConfigurationScriptNode1’, u’sceneConfigurationScriptNode2’,
u’uiConfigurationScriptNode1’]

Context:

<Sgtk Context: Project: {‘type’: ’Project’, ’name’: [redacted] ’id’: 91}
Entity: {‘type’: ’Asset’, ’name’: ’prSteamCharabanc01’, ’id’: 1955}
Step: {‘type’: ’Step’, ’name’: ’modLow’, ’id’: 175}
Task: {‘type’: ’Task’, ’name’: ’modLow’, ’id’: 9113}
User: {‘type’: ’HumanUser’, [redacted] ’name’: [redacted]
Shotgun URL: [redacted]
Additional Entities: []
Source Entity: {‘project’: [redacted] ’type’: ’Task’, [redacted]>
```

Required technology stack

- Programming Skills:
  - Databases (shotgun, ftrack,..)
  - Software Engineering principles
  - User Interface and User Experience skills
- Programming languages:
  - Python
  - DCC (Digital Content Creation Tool) specific programming languages
  - C++
- DCC Skills:
  - 3D tools (maya, 3dsmax, cinema4d, blender, houdini, substance, unreal engine..)
  - 2D tools (adobe suite, nuke,...)
- Nice to have skills:
  - Web-development -> visualization

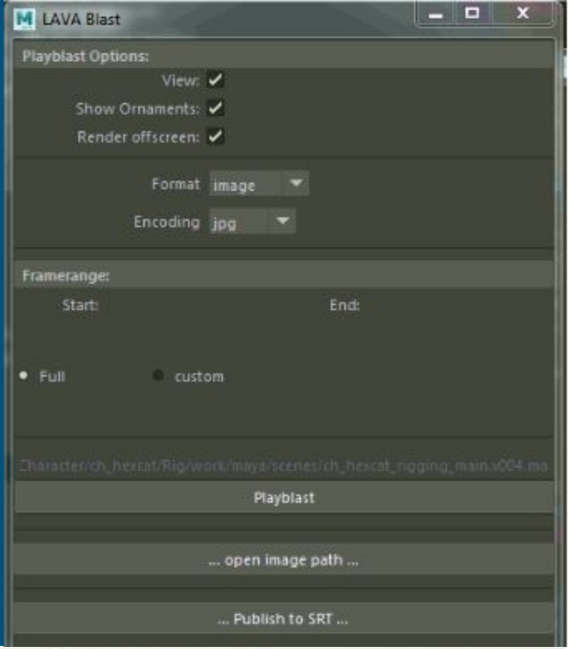
Creating GUI for Maya

- Options for creating GUI in Maya:
  - Maya build in window class
  - Tkinter
  - QT, pyside, pyqt, qtpy
  - Kivy
  - ...

Creating GUI for Maya

- Maya build in window class
  - Window class to hold your widgets in
    - cmds.window()
  - Layout to define how your widgets are displayed
    - cmds.rowLayout()
    - cmds.columnlayout()
    - cmds.rowColumnLayout()
  - The widgets you want to use
    - cmds.text()
    - cmds.button()
  - Connect your widgets to their parent
    - cmds.setParent()
  - cmds.showWindow()

See example:
https://github.com/JanPhKoch/JPK-PythonForMaya/blob/master/examples/asset_duplicator_completed.py


LAVA Blast

Playblast Options:

View: ☑

Show Ornaments: ☑

Render offscreen: ☑

Format image

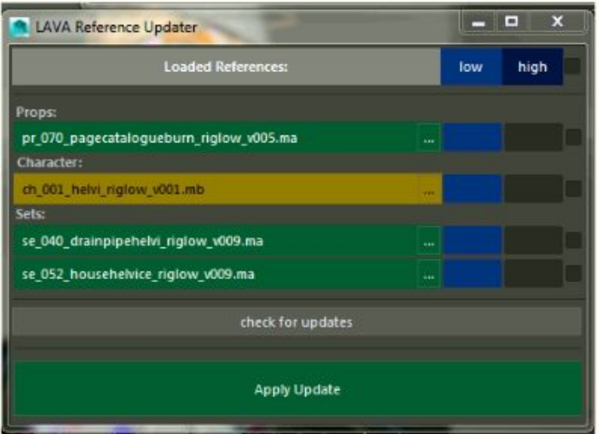
Encoding jpg

Framerange:

Start: End:

● Full ○ custom

Character/ch_hexcat/Rig/work/maya/scenes/ch_hexcat_rigging_main.v004.ma


LAVA Reference Updater

Loaded References: low high

Props:

pr_070_pagecataloguebum_riglow_v005.ma

Character:

ch_001_helvi_riglow_v001.mb

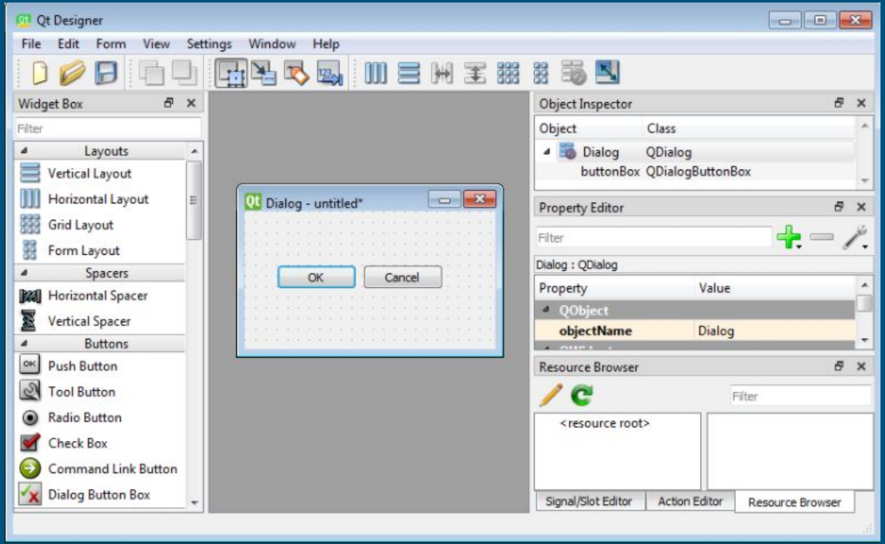
Sets:

se_040_drainpipehelvi_riglow_v009.ma

se_052_househelvice_riglow_v009.ma

Creating GUI for Maya

- QT (c++ platform independent GUI toolkit)
- PYQT, Pyside, QTPY
  - Python implementations of QT (so you don’t need to build your interfaces in c++)
- QT Designer drag-and-drop tool to quickly build your GUI (link to manual)



Object	Class
Dialog	QDialog
buttonBox	QDialogButtonBox

Dialog : QDialog

Property	Value
QObject	
objectName	Dialog

Creating GUI for Maya

- Create your own QT User Interface!
  - Open the QT Designer (under the maya install: C:\Program Files\Autodesk\Maya2020\bin)
  - Create a MainWindow
  - Drag ‘n’ drop widgets into your window
  - Connect them to your maya python code

Error/exception handling

Error/exception handling

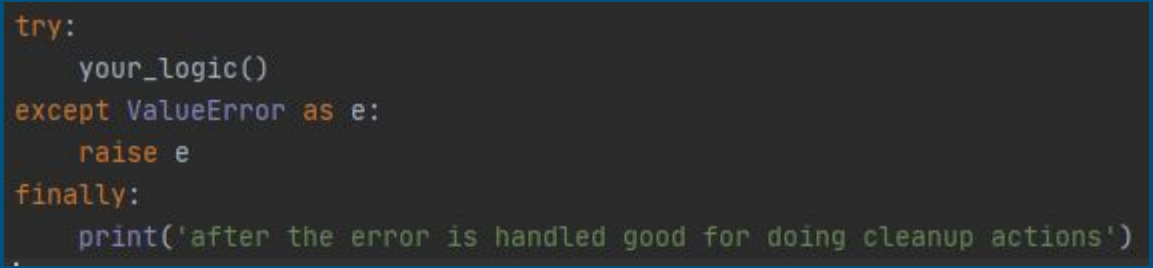
- The process of handling problems before they happen
  - Exp: user is required to input digits(integer) values but inputs “abc” (string), your tool requires integers otherwise your multiplication function crashes your tool.
- Try to verify input to prevent issues/hard crashes of your tools and wrong outcomes
- The programmer usually can understand crashes/error messages from the debugger and knows what to do to prevent them, the user not!
- Present the user with easily understandable error messages

Error/exception handling

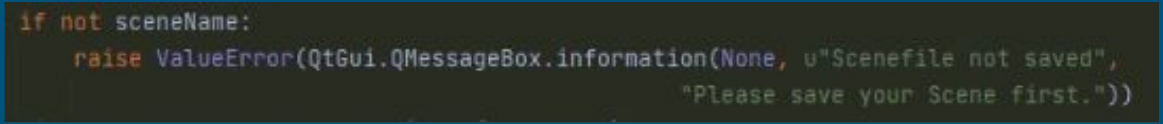
- Tools for error handling
  - Try, except, finally
    - Try (your code that you want to catch specific errors in)
    - Except (the error type you want to handle)
    - Finally (optional cleanup functionality)

Error/exception handling

- Tools for error handling
  - Try, except, finally

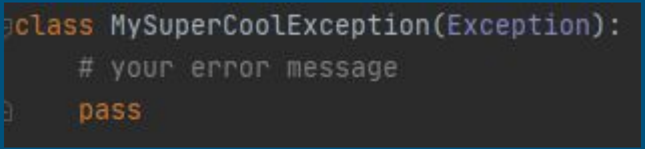


```
try:
    your_logic()
except ValueError as e:
    raise e
finally:
    print('after the error is handled good for doing cleanup actions')
```

  - raise errors
    - 

```
if not sceneName:
    raise ValueError(QtGui.QMessageBox.information(None, u"Scenefile not saved",
                                                   "Please save your Scene first."))
```

Error/exception handling

- Always check the build in exception list first: https://docs.python.org/3.7/library/exceptions.html
- If you want to throw and specific exception:
  - Derive from Exception base class:
    - 

```python
class MySuperCoolException(Exception):
    # your error message
    pass
```

Creating Maya Plugins

- What are Maya Plugins?
- When should I create Plugins?
- How to create them?
- Example

Creating Maya Plugins

- What are Maya Plugins?
  - Maya Software Extensions written in c++
    - File translator
    - New node objects
    - New MEL & Python commands
  - OS dependent file extensions
    - Windows: .mll
    - Linux: .so
    - Mac: .bundle

Creating Maya Plugins

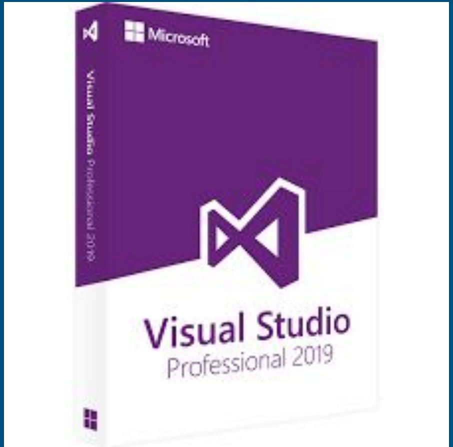
- When should I create Plugins?
  - Plugins code should have a higher code quality standard then .mel .py scripts.
  - Finished and clear features that do not change regularly.

Good Plugin examples:
 - file handles for exporting (.abc, .fbx)
 - node creation for pipeline stability

Creating Maya Plugins

How to create them?

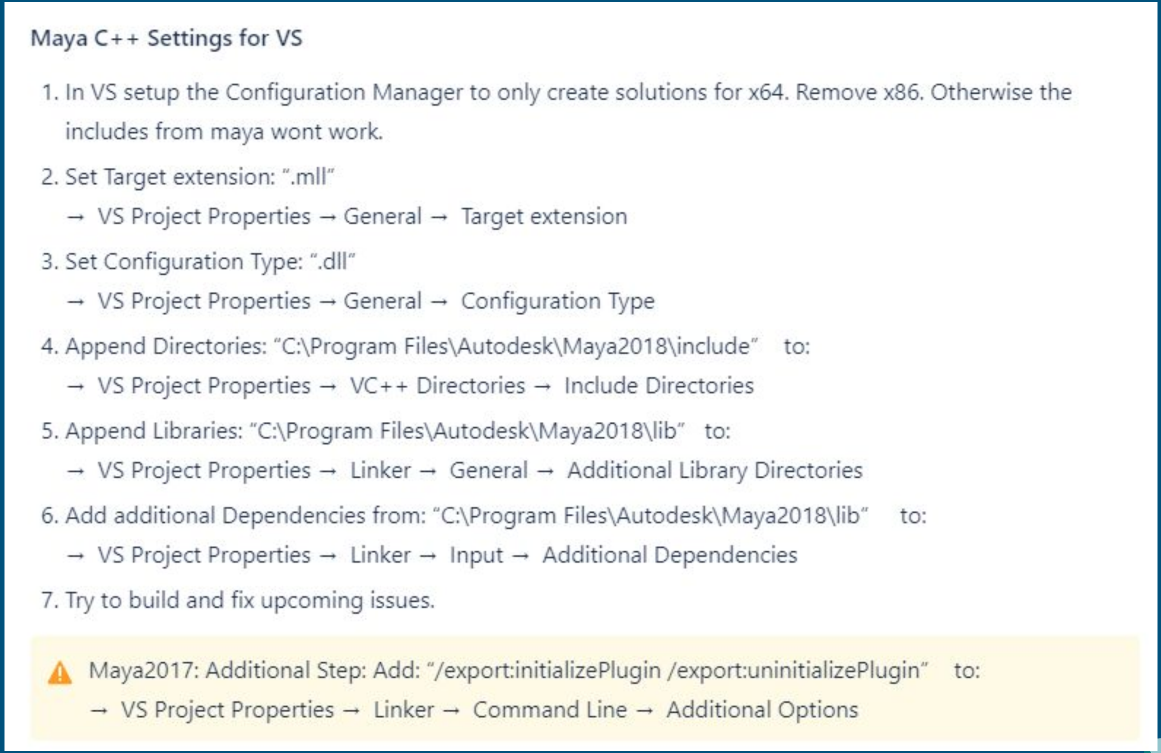
1. Install Visual Studio (IDE)
   1.1. Check which version of VS you need for which maya version. Maya 2018 = Visual Studio 2015 (link)



Creating Maya Plugins

How to create them?

1. Install Visual Studio (IDE)
2. VS Project
   (see settings on attached image
   for compiling .mll (windows) maya plugins -->)
3. Compile your VS Project



Maya C++ Settings for VS

1. In VS setup the Configuration Manager to only create solutions for x64. Remove x86. Otherwise the includes from maya wont work.
2. Set Target extension: ".mll"
   → VS Project Properties → General → Target extension
3. Set Configuration Type: ".dll"
   → VS Project Properties → General → Configuration Type
4. Append Directories: "C:\Program Files\Autodesk\Maya2018\include" to:
   → VS Project Properties → VC++ Directories → Include Directories
5. Append Libraries: "C:\Program Files\Autodesk\Maya2018\lib" to:
   → VS Project Properties → Linker → General → Additional Library Directories
6. Add additional Dependencies from: "C:\Program Files\Autodesk\Maya2018\lib" to:
   → VS Project Properties → Linker → Input → Additional Dependencies
7. Try to build and fix upcoming issues.

⚠ Maya2017: Additional Step: Add: "/export:initializePlugin /export:uninitializePlugin" to:
→ VS Project Properties → Linker → Command Line → Additional Options

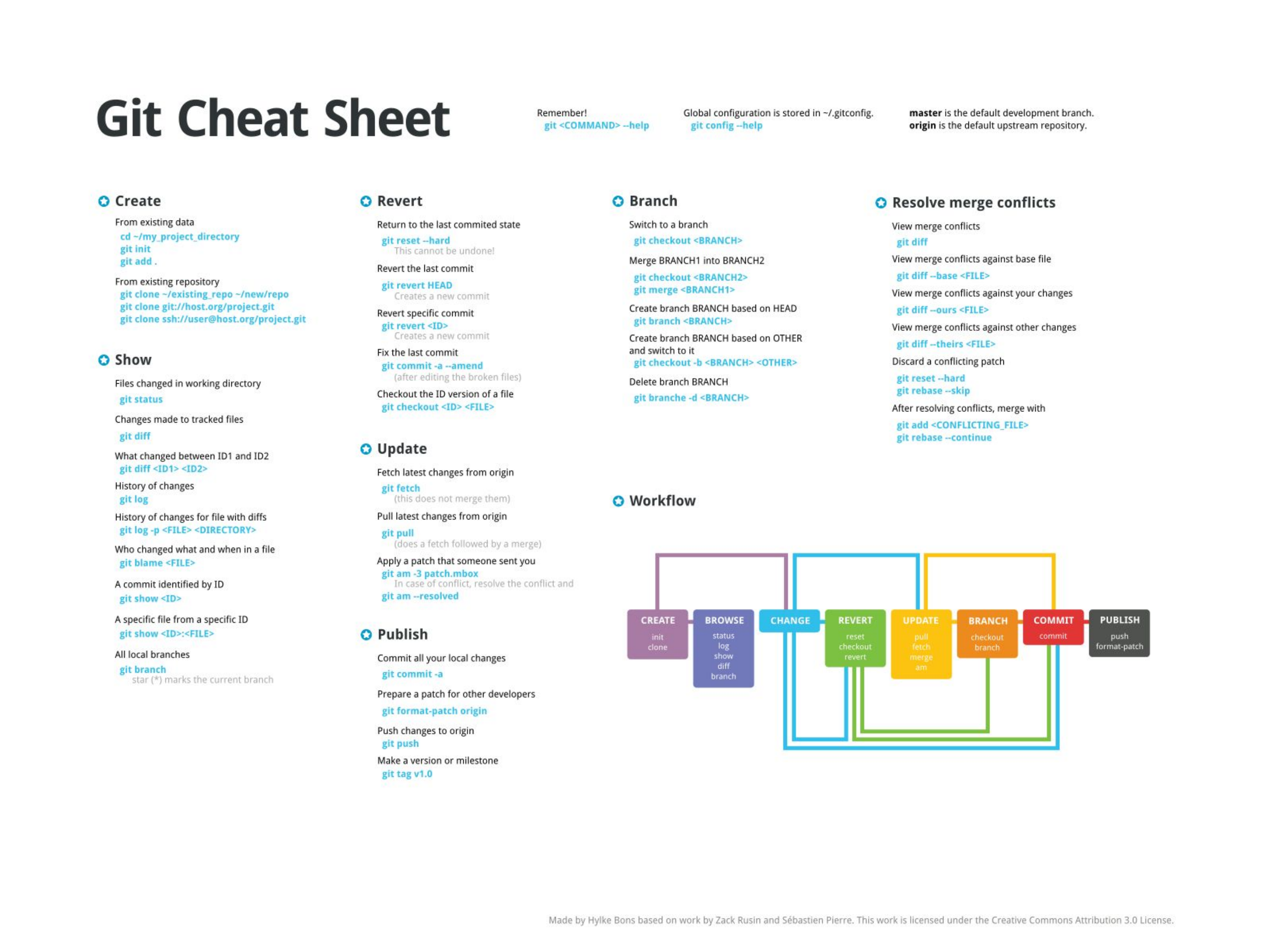
Version Control (Git)

- ???

Version Control (Git)

- https://git-scm.com/ (download git + find the docs here)
- Management of multiple Versions of the same file.
- Clearer and quicker version switches because of “comments” that describe specific changes
- Allows developing with several developers on one file
- Clearer for others to understand changes you made

! Take a look at the .git folder of the course repository (JPK-PythonForMaya) !



Git Cheat Sheet

Remember!

```
git <COMMAND> --help
```

Global configuration is stored in ~/.gitconfig.

```
git config --help
```

master is the default development branch.
origin is the default upstream repository.

Create

From existing data

```
cd ~/my_project_directory
git init
git add .
```

From existing repository

```
git clone ~/existing_repo ~/new/repo
git clone git://host.org/project.git
git clone ssh://user@host.org/project.git
```

Show

Files changed in working directory

```
git status
```

Changes made to tracked files

```
git diff
```

What changed between ID1 and ID2

```
git diff <ID1> <ID2>
```

History of changes

```
git log
```

History of changes for file with diffs

```
git log -p <FILE> <DIRECTORY>
```

Who changed what and when in a file

```
git blame <FILE>
```

A commit identified by ID

```
git show <ID>
```

A specific file from a specific ID

```
git show <ID>:<FILE>
```

All local branches

```
git branch
```

star (*) marks the current branch

Revert

Return to the last commited state

```
git reset --hard
```

This cannot be undone!

Revert the last commit

```
git revert HEAD
```

Creates a new commit

Revert specific commit

```
git revert <ID>
```

Creates a new commit

Fix the last commit

```
git commit -a --amend
```

(after editing the broken files)

Checkout the ID version of a file

```
git checkout <ID> <FILE>
```

Update

Fetch latest changes from origin

```
git fetch
```

(this does not merge them)

Pull latest changes from origin

```
git pull
```

(does a fetch followed by a merge)

Apply a patch that someone sent you

```
git am -3 patch.mbox
```

In case of conflict, resolve the conflict and

```
git am --resolved
```

Publish

Commit all your local changes

```
git commit -a
```

Prepare a patch for other developers

```
git format-patch origin
```

Push changes to origin

```
git push
```

Make a version or milestone

```
git tag v1.0
```

Branch

Switch to a branch

```
git checkout <BRANCH>
```

Merge BRANCH1 into BRANCH2

```
git checkout <BRANCH2>
git merge <BRANCH1>
```

Create branch BRANCH based on HEAD

```
git branch <BRANCH>
```

Create branch BRANCH based on OTHER and switch to it

```
git checkout -b <BRANCH> <OTHER>
```

Delete branch BRANCH

```
git branche -d <BRANCH>
```

Resolve merge conflicts

View merge conflicts

```
git diff
```

View merge conflicts against base file

```
git diff --base <FILE>
```

View merge conflicts against your changes

```
git diff --ours <FILE>
```

View merge conflicts against other changes

```
git diff --theirs <FILE>
```

Discard a conflicting patch

```
git reset --hard
git rebase --skip
```

After resolving conflicts, merge with

```
git add <CONFLICTING_FILE>
git rebase --continue
```

Workflow

CREATE
init
clone

BROWSE
status
log
show
diff
branch

CHANGE

REVERT
reset
checkout
revert

UPDATE
pull
fetch
merge
am

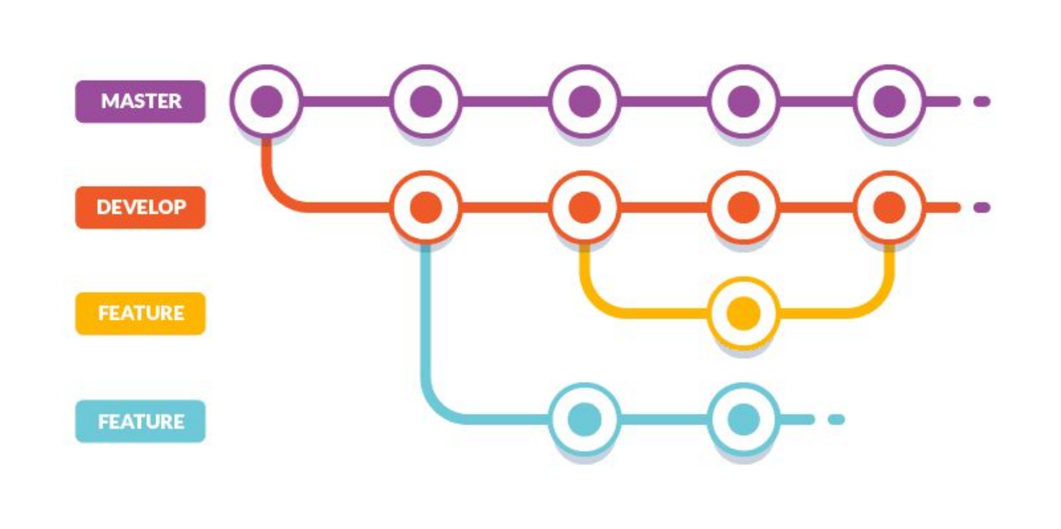
BRANCH
checkout
branch

COMMIT
commit

PUBLISH
push
format-patch

Version Control (Git)

- Feature-Branch Workflow



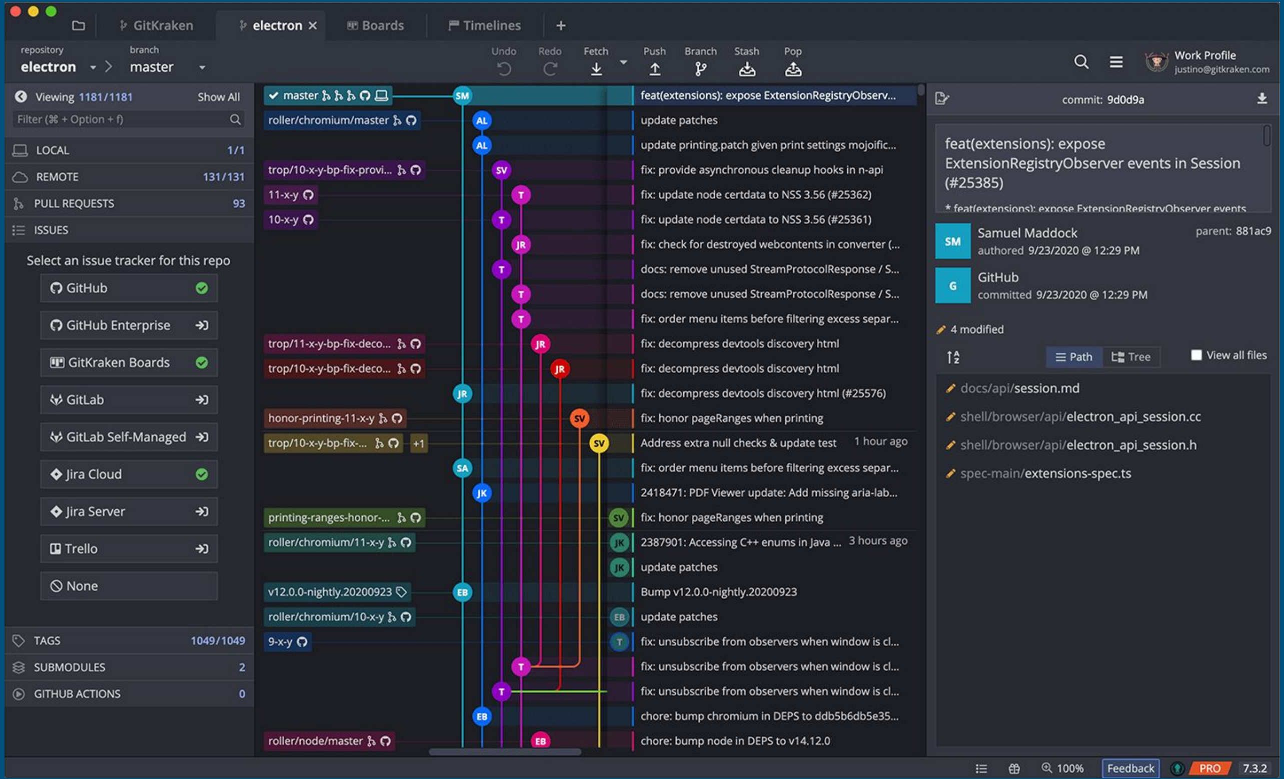
Version Control (Git)

- Github + Gitlab
    - Online Storage for your dev-projects. (remote repository)
    - Push your local development to your remote repo
    - Pull + Merge requests, Issues
    - Work collaboratively
    - Most Open Source software projects are managed either on Github or Gitlab.

Version Control (Git)

- GUI
  - Quicker to get an overview about all feature branches
  - Easier, prettier
- Most-used GUI for git
  - GitGui: https://git-scm.com/book/en/v2/Appendix-A%3A-Git-in-Other-Environments-Graphical-Interfaces
  - Gittower: https://www.git-tower.com/
  - Gitkraken: https://www.gitkraken.com/
  - Github Desktop: https://desktop.github.com/

Version Control (Git)



master — SM — feat(extensions): expose ExtensionRegistryObserv...
roller/chromium/master — AL — update patches
AL — update printing.patch given print settings mojoific...
trop/10-x-y-bp-fix-provi... — SV — fix: provide asynchronous cleanup hooks in n-api
11-x-y — T — fix: update node certdata to NSS 3.56 (#25362)
10-x-y — T — fix: update node certdata to NSS 3.56 (#25361)
JR — fix: check for destroyed webcontents in converter (...
T — docs: remove unused StreamProtocolResponse / S...
T — docs: remove unused StreamProtocolResponse / S...
T — fix: order menu items before filtering excess separ...
trop/11-x-y-bp-fix-deco... — JR — fix: decompress devtools discovery html
trop/10-x-y-bp-fix-deco... — JR — fix: decompress devtools discovery html
JR — fix: decompress devtools discovery html (#25576)
honor-printing-11-x-y — SV — fix: honor pageRanges when printing
trop/10-x-y-bp-fix-... +1 — SV — Address extra null checks & update test — 1 hour ago
SA — fix: order menu items before filtering excess separ...
JK — 2418471: PDF Viewer update: Add missing aria-lab...
printing-ranges-honor-... — SV — fix: honor pageRanges when printing
roller/chromium/11-x-y — JK — 2387901: Accessing C++ enums in Java ... — 3 hours ago
JK — update patches
v12.0.0-nightly.20200923 — EB — Bump v12.0.0-nightly.20200923
roller/chromium/10-x-y — EB — update patches
9-x-y — T — fix: unsubscribe from observers when window is cl...
T — fix: unsubscribe from observers when window is cl...
T — fix: unsubscribe from observers when window is cl...
EB — chore: bump chromium in DEPS to ddb5b6db5e35...
roller/node/master — EB — chore: bump node in DEPS to v14.12.0

commit: 9d0d9a

feat(extensions): expose ExtensionRegistryObserver events in Session (#25385)

* feat(extensions): expose ExtensionRegistryObserver events

SM Samuel Maddock
authored 9/23/2020 @ 12:29 PM
parent: 881ac9

G GitHub
committed 9/23/2020 @ 12:29 PM

4 modified

Path Tree ☐ View all files

docs/api/session.md
shell/browser/api/electron_api_session.cc
shell/browser/api/electron_api_session.h
spec-main/extensions-spec.ts

Version Control (Git)

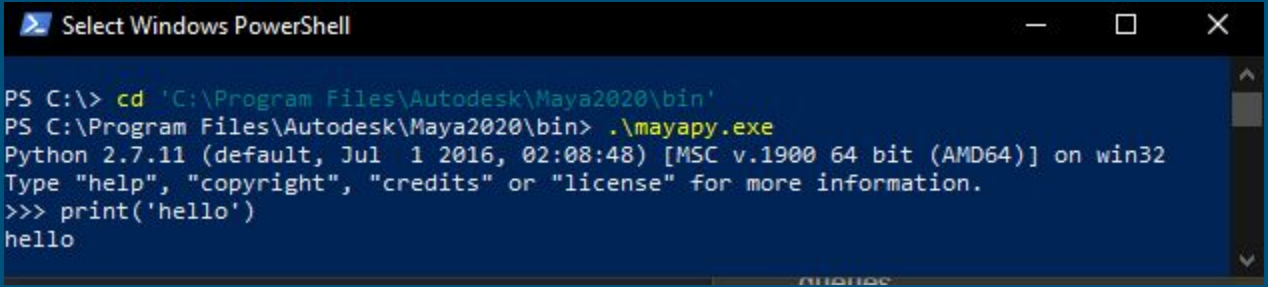
1. Create Github Account
2. Download + Install Gitkraken or github desktop
3. Create your local project
4. Push your project to Github
5. Make some changes and Push again

Bonus: Invite your classmates to your repo and let them make a request!

Time: ca. 20 min

Maya Standalone (GUI-less maya)

- Location:
  - Install directory: C:\Program Files\Autodesk\Maya2020\bin\mayapy.exe
- How to run (shell, console, terminal):
  - cd 'C:\Program Files\Autodesk\Maya2020\bin'
  - .\mayapy.exe



```
PS C:\> cd 'C:\Program Files\Autodesk\Maya2020\bin'
PS C:\Program Files\Autodesk\Maya2020\bin> .\mayapy.exe
Python 2.7.11 (default, Jul  1 2016, 02:08:48) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello')
hello
```

Maya Testing

- Common tests:
  - Unit (test each unit on their own not the whole tool at once)
  - User (let others test your tools)
  - Acceptance (are the inputs correct and are they correctly managed?)
- Testing environment
  - Mockup = your virtual test objects (so you don't need to change actual data)
  - Keep testscenes simple + clean
- Test driven development (TDD)
  - write and organize your software probably from the start
  - Write tests first before your actual logic
  - Powerful tool to help you keep on track and quickly find out about new issues when updating your software

Maya Testing

- Maya Testing Framework by Chad Vernon

  For testing your code with .mayapy without opening a maya scene.

  - https://github.com/chadmv/cmt
  - http://www.chadvernon.com/blog/unit-testing-in-maya/

Thank you for your attention

- Course Repository: https://github.com/JanPhKoch/JPK-PythonForMaya
- Contact me: Slack or Mail: janphkoch@gmail.com