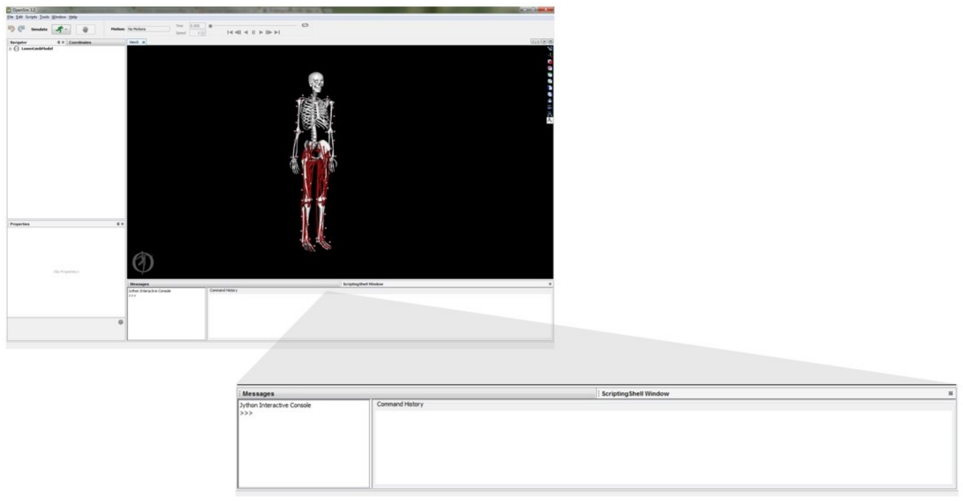
Scripting in the GUI

As of version 3.0, OpenSim comes with a built in scripting shell that allows user access to the OpenSim API for loading, editing, and building models, running tools, plotting results, and more. The syntax for the scripting shell in the GUI is Python.

What's Available?

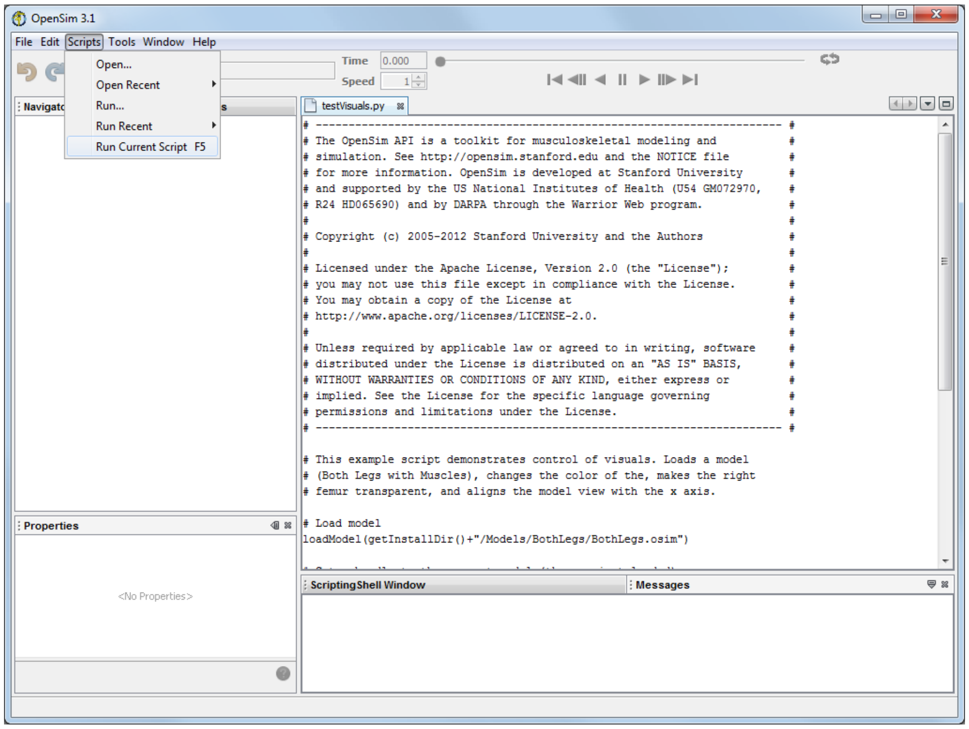


- Access to the API for model editing and building
- Access to the tools for processing and workflow batch processing
- Access to most of the commands available from the OpenSim application menu and most of the selections tasks you can currently perform in the GUI
- A streamlined interface to the plotter so that you can easily create, customize and export curves from the plotter.
- Limited access to the graphics window (e.g. selection and camera control)

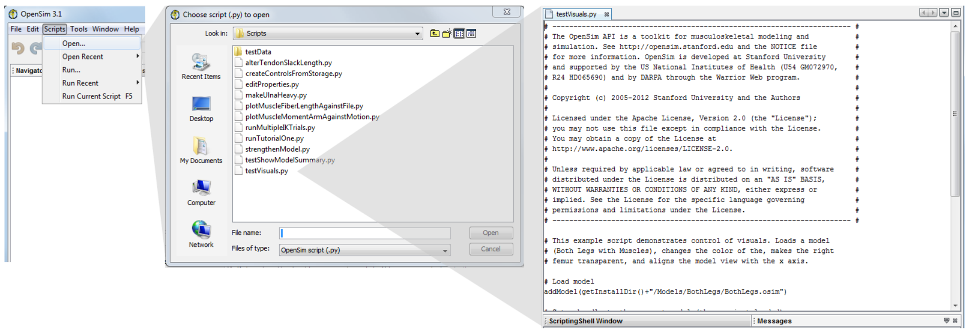
Getting Started

1. Launch OpenSim and select **Window > ScriptingShell** to bring up the scripting window. This will display the main scripting shell window ("Jython Interactive Console") and the "Command History" of executed scripting commands.

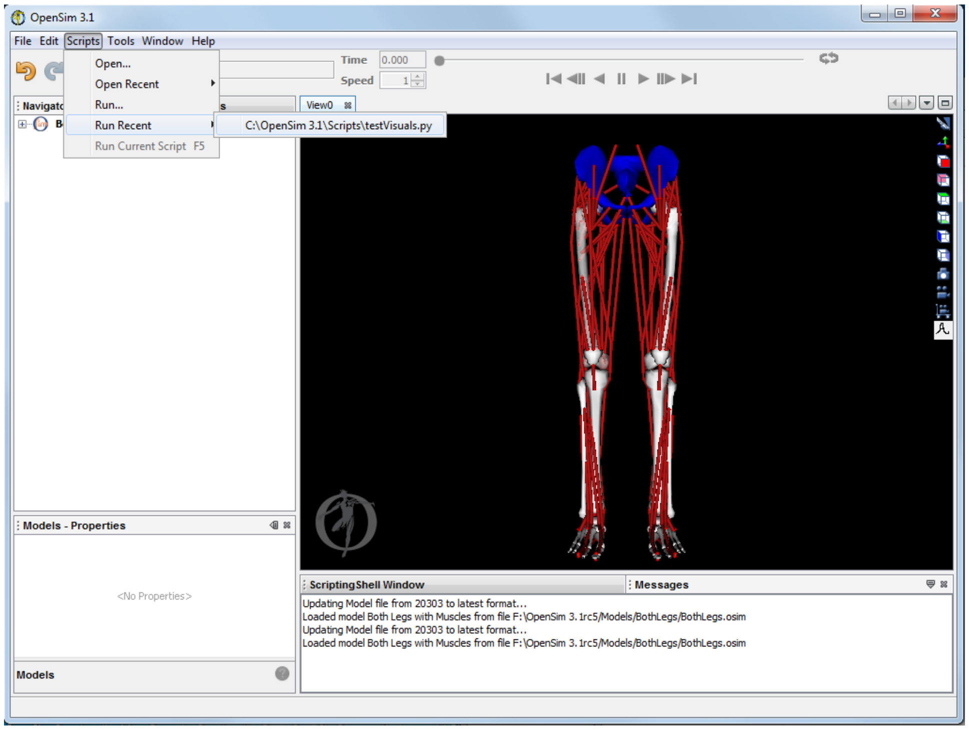
2. Select **Scripts > Open**.... Find the Scripts folder located in the directory where you installed OpenSim. There are many example scripts included with the OpenSim distribution. You can review these to get an idea of the capabilities of scripting or you can modify them to streamline the workflow of your own research.



```
# ------------------------------------------------------------------------ #
# The OpenSim API is a toolkit for musculoskeletal modeling and           #
# simulation. See http://opensim.stanford.edu and the NOTICE file         #
# for more information. OpenSim is developed at Stanford University       #
# and supported by the US National Institutes of Health (U54 GM072970,    #
# R24 HD065690) and by DARPA through the Warrior Web program.             #
#                                                                         #
# Copyright (c) 2005-2012 Stanford University and the Authors             #
#                                                                         #
# Licensed under the Apache License, Version 2.0 (the "License");         #
# you may not use this file except in compliance with the License.        #
# You may obtain a copy of the License at                                 #
# http://www.apache.org/licenses/LICENSE-2.0.                             #
#                                                                         #
# Unless required by applicable law or agreed to in writing, software     #
# distributed under the License is distributed on an "AS IS" BASIS,       #
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or         #
# implied. See the License for the specific language governing            #
# permissions and limitations under the License.                          #
# ------------------------------------------------------------------------ #

# This example script demonstrates control of visuals. Loads a model
# (Both Legs with Muscles), changes the color of the, makes the right
# femur transparent, and aligns the model view with the x axis.

# Load model
loadModel(getInstallDir()+"/Models/BothLegs/BothLegs.osim")
```



```
# ------------------------------------------------------------------------ #
# The OpenSim API is a toolkit for musculoskeletal modeling and           #
# simulation. See http://opensim.stanford.edu and the NOTICE file         #
# for more information. OpenSim is developed at Stanford University       #
# and supported by the US National Institutes of Health (U54 GM072970,    #
# R24 HD065690) and by DARPA through the Warrior Web program.             #
#                                                                         #
# Copyright (c) 2005-2012 Stanford University and the Authors             #
#                                                                         #
# Licensed under the Apache License, Version 2.0 (the "License");         #
# you may not use this file except in compliance with the License.        #
# You may obtain a copy of the License at                                 #
# http://www.apache.org/licenses/LICENSE-2.0.                             #
#                                                                         #
# Unless required by applicable law or agreed to in writing, software     #
# distributed under the License is distributed on an "AS IS" BASIS,       #
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or         #
# implied. See the License for the specific language governing            #
# permissions and limitations under the License.                          #
# ------------------------------------------------------------------------ #

# This example script demonstrates control of visuals. Loads a model
# (Both Legs with Muscles), changes the color of the, makes the right
# femur transparent, and aligns the model view with the x axis.

# Load model
addModel(getInstallDir()+"/Models/BothLegs/BothLegs.osim")
```



```
Updating Model file from 20303 to latest format...
Loaded model Both Legs with Muscles from file F:\OpenSim 3.1rc5/Models/BothLegs/BothLegs.osim
Updating Model file from 20303 to latest format...
Loaded model Both Legs with Muscles from file F:\OpenSim 3.1rc5/Models/BothLegs/BothLegs.osim
```

3. Choose "testVisuals.py". The script will open in the OpenSim GUI's file editor. This script demonstrates how you can control visualization via scripting. The script loads a model ("gait2392_simbody"), changes the color of the pelvis, makes the right femur transparent, and aligns the model view with the x axis.

4. To run the script **hit F5**, with the script open. Alternately, you can choose **Scripts > Run Current Script**.

5. Go to the **Scripting Shell Window** and review the **Comm and History**, on the right hand side of the window. This will show you all of the scripting commands that were executed. Review the commands that performed each of the actions in the testVisuals script.

6. Try executing another script command to change the opacity or color of the pelvis. Enter the commands at the prompt >>>. You can copy and paste lines of code to test individual commands.

7. Try doing the same thing by modifying the testVisuals.py script. Find the "Scripts" folder in your OpenSim Install directory, open the file with a text editor (e.g. Notepad++), and add the command. Save the script with a new name (e. g. myTestVisuals.py). Return to OpenSim and try out the new script. You can run the script directly by selecting **Scripts -> Run...** and navigating to your script. Does it perform the function you expect?

8. Now try running, reviewing, and modifying the other example scripts included with your OpenSim distribution. The full list is below.

9. Learn more about scripting in the section on Common Scripting Commands.

Tips and Tricks

- You can change the location of the Scripts directory by going to **Edit > Preferences...** and changing the directory under the "Path: Scripts Path" option (see User Preferences for more information).
    - Note that you have to restart OpenSim for the change to take place.
    - The default value is **<ResourcesDir>/Code/GUI**, and the value is reset if you run **installResources()** in the ScriptingShell.
    - In the ScriptingShell, you can access the value of this preference using **getScriptsPath()**.
- After loading a script, variables defined in the script are available for future reference.
- A panel to echo commands and files executed in the scripting shell is available (the "Command History"). You can clear the command history by right clicking in the window and selecting "Clear".
- If there are errors in running a script from file, you can view the corresponding error messages by clicking on the red error icon in the bottom right of the OpenSim screen, and then selecting "Show Details"
- In the GUI scripting shell, code executed as part of for loops or if statements is controlled by indentation, rather than braces as in C++ or the for /end convention in Matlab. Otherwise, at the ScriptingShell command prompt, you must **enter one line at a time.**

Interacting with the GUI (including the Plotter)

In the GUI scripting shell, we've provided a set of commands that allow you to easily perform some basic plotting functionality. This includes adding a plot window, adding curves, changing the legend, plotting from file, exporting the plot, etc.

https://simtk.org/api_docs/opensim/gui_docs/org/opensim/console/OpenSimPlotter.html

These commands are exercised in the examples "plotMuscleFiberLengthAgainstFile.py" and "plotMuscleMomentArmAgainstMotion.py".

There are additional advanced commands that you can call (the ones used by the GUI itself), though these commands have not been fully documented and tested with the scripting functionality, so please use with caution.

https://simtk.org/api_docs/opensim/gui_docs/org/opensim/console/gui.html

Available Example Scripts

Script Name	Description
runTutorialOne.py	This example script performs the steps performed in Tutorial 1 - Intro to Musculoskeletal Modeling.
runTutorialTwo.py	This example script performs the steps performed in Tutorial 2 - Simulation and Analysis of a Tendon Transfer Surgery.
runTutorialThree.py	This example script performs the steps performed in Tutorial 3 - Scaling, Inverse Kinematics, and Inverse Dynamics.
testShowModelSummary.py	Displays information about the current model in a standalone dialog.
plotMuscleFiberLengthAgainstFile.py	Shows how to create and display a plot window. The script loads the BothLegs OpenSim model and adds curves of fiber length for the model. Then, it loads and plots data from a storage file, which contains fiber lengths for the model Subject01_simbody that is included with the OpenSim distribution.
plotMuscleMomentArmAgainstMotion.py	Shows how to plot muscle moment arms as a function of a motion.
makeUlnaHeavy.py	Shows how to create a modified version of a model that is loaded in the GUI. The script increases the mass of the ulna. The modified model is then loaded in the GUI.
alterTendonSlackLength.py	Shows how to change the attributes of the muscles for the current model.
runScaling.py	Runs the scale tool for the Gait2354 model, and loads the generic and scaled models in the GUI.
runMultipleIKTrials.py	Runs multiple inverse kinematics trials. To see the results, load the model and IK output in the GUI.
strengthenModel.py	Increases the max. isometric force of all the muscles in the currently loaded model. A pop-up dialog displays a confirmation with the name of the new model.
editProperties.py	Script to demonstrate how to edit model component properties.
ModelBuilder.py	Helper functions for adding specific types of components (Bodies, Frames, Joints, Geometry) to a model.
createControlsFromStorage.py	Helper function to create a controls.xml file from a .sto file.

Next: Scripting with Matlab

Previous: Scripting Versions of OpenSim C++ API Calls

Home: Scripting and Development