## CONTENTS

## Chapter 1:

- Introduction to Nuke's interface and API;
- Basics on node creation via Python;
- Printing a node's class;
- Setting node's knobs values;
- Different methods for creating nodes: '**nuke.createNode'** and '**nuke.nodes.*nodename';***
- Setting knobs' values: the **'setValues'** command;
- Setting nodes' values and labels;
- Setting colors and the use of a standard palette;
- The '**getColor()'** method to pick colors from the color palette;
- The '**tile_color'** knob and smart methods to figure out color values;
- The use of variables;
- Methods to call a variable throughout a program;
- The '**nuke.root()'** knobs and methods to set the first and the last frame of your comp script programmatically: '**nuke.root()['first_frame'].setValue(...)'** and '**nuke.root().firstFrame()';**
- Operations on the Format node;
- The use of html tags to change the appearance of Nuke's nodes;
- Adding custom knobs on standard Nuke's nodes;
- The use of '**setExpression()'** to set expressions on nodes;
- Basic methods for knobs creation.

## Chapter 2:

- Techniques for creating arrays of nodes;
- Method for the node's creation: the '**inpanel**' switch;
- The use of '**nuke.message'** to display pop up messages;
- Introducing animation basics;
- The '**setValueAt()**' and '**setAnimated()**' methods;
- Methods to set animated keys on knobs;
- Methods for connecting nodes' slots programmatically;
- The '**nuke.zoom'** and **'nuke.zoomToFitSelected()'** commands to zoom on selected nodes or groups of nodes;
- The '**nukescripts.remove_inputs()'** method to disconnect all node's input slots for specific nodes;
- First steps with the **menu.py** file - Creating basic Nuke's menus;
- Basics on how to set default values for knobs: the '**knobDefaults()'** method;
- A practical example on how to use the **NoOp** node to create your own gizmo and run Python programs from it;
- Method to create a new Format programmatically;
- Loops and iteration operators: The "**for**" loop and the importance of the indentation;
- First basics on timeline management: the **'nuke.frame()**' command;
- Using "**for**" loops for setting values on multiple nodes;
- The '**hideControlPanel()'** method to hide control panels;
- Technique to get a user-defined color's value;

## Chapter 3:

- Methods for arranging comp script's nodes: “**autoplace**” and ‘**autoplacesnap**’;
- Setting nodes' positions in the Node graph programmatically;
- Creating ‘**nuke.message**’ pop-ups using different colors and fonts’ sizes;
- Deleting nodes: the ‘**nuke.delete()**’ command;
- Methods for extracting nodes and groups from the script's three: ‘**nuke.extractSelected()’**
- Working with Groups;
- Creating or displacing Nuke’s groups: ‘**nuke.makeGroup()**’;
- Clearing animations on nodes’s knobs: ‘**clearAnimated()**’;
- Working examples with ‘**dependencies**’ and ‘**dependent**’ nodes;
- Methods for inverting selections using ‘**nuke.selectAll()**’ and ‘**nuke.invertSelection()**’;
- The “**len**” method;
- Examples with the '**len**' command and ‘**nuke.message**’ ;
- **LISTS –** methods for creating and manage lists of objects;
- Managing list indexes by using the **%** operator and the ‘**.format**’ method;
- Operation on lists;
- The ‘**reverse()**’ method for lists;
- Ordering list objects with The ‘**list.sort()**’ method;
- Techniques to merge lists together;
- Experimenting with lists: The creation of a 'selection saver';
- Digits, numbering and the ‘**zfill**’ method;
- The '**join**' method on strings;
- Different types of lists: ‘**Tuples**’ and ‘**Sets**’;
- RGB to HEX conversion method.

## Chapter 4:

- The use of the ‘**getInput**’ command to input user’s values;
- Working with Channels - adding new channels - examples;
- The ‘**nuke.Layer()**’ command;
- The “**range**” loop and methods for creating multiple nodes;
- Using the '**range**' loop for rearranging nodes’ inputs;
- The Python ‘**exec()**’ command;
- Experiments with loops and lists: A program to create 3D cards with the ‘**range**’ loop;
- The “**range**” loop and the **random** module for creating a busy 3D “traffic” scene;
- The ‘**iter()**’ method for loops;

## Chapter 5:

- Creating an array of Cameras using a loop and math functions;
- Commands for copying nodes: ‘**nuke.forceClone()**’ and ‘**nuke.cloneSelected()**’;
- Using the ‘**random**’ method for cloning nodes ;
- Creating Backdrops programmatically: **‘nukescripts.autoBackdrop()’;**
- Programmatically copying and pasting nodes in Nuke: **‘nuke.nodeCopy()**’ and **‘nuke.NodePaste()**’;
- Example program: a simple method for turning all backdropNodes' colors to a specific one;
- Methods for collapsing nodes in a group: ‘**nuke.collapseToGroup**’;
- Accessing Groups contents: the ‘**begin()**’ and ‘**end()**’ methods as opposed to the ‘**with()**’ method;

- Programmatic creation of a special stickyNote with nodes' names inside a collapsed Group;
- Inspecting a Group content: the '**nuke.showDag**' command;
- How to expand groups: '**nuke.expand**' and '**nuke.expandSelectedGroup()**';
- Basics on the conditional operators: "**if**" .

## Chapter 6:

- Managing Nuke's plugins with the '**nuke.plugins**' method;
- A technique to check if a gizmo is not a standard Nuke's gizmo;
- Working with node's classes – an example on how to get rid of objects;
- Further words on conditional operators and nested loops;
- Animation commands to set animation keys and perform checks on knobs: '**isKey()**' and '**isAnimated()**';
- Method for getting a knob's value at a specific frame: '**getValueAt()**';
- The '**copy**' module and the '**copy()**' command;
- Basics on Error handling and debugging;
- A method to check if a node *has an expression;*
- Working with the backdrops' **z_order** knob;
- Resetting parameters: examples with '**unsplitView()**' on stereo compositing scripts;
- The use of the '**Try/Except**' method and the error handling;
- Working with floating and integer numbers;
- The '**isinstance()**' method;
- Method for programmatically splitting Stereo views;
- Example: a program to set a '**$gui**' expression for all the selected nodes and for checking on any errors;
- How to ask the user what to do on a choice : the '**nuke.ask**' command;
- Setting nodes to their default color values: '**nuke.defaultNodeColor()**';
- setting nodes' knobs to their default values: the '**defaultValue()**' method;

## Chapter 7:

- The '**min()**' and '**max()**' methods;
- Example program: Creating a tool to check on all the script's nodes if the main input slot is connected and to put a BackdropNode behind them otherwise;
- The '**maxInputs()**' method to determine the number of a node's inputs;
- Further programming experiences with loops on backdropNodes;
- Checking method for the existence of a node with a given name using a *boolean* '**for**' loop ;
- Example: printing a warning message if a node doesn't get any input;
- The '**nuke.critical**' pop-up message;
- A method for programmatically check the image's Bbox size of a Viewer;
- Further operations on the timeline with '**nuke.frame()**' and the '**frameRange**' method;
- A programming example on the timeline: how to set keys for a specific knob every 10 frames;
- Programming expressions for creating animation curves;
- Writing expressions for animations : baking, translating, scale and invert an animated curve;
- The '**nuke.animation**' command;
- The use of '**isupper()**' . '**upper()**' and '**lower()**' methods to change the capital letters in a string;
- The '**capitalize()**' method
- Operations on baked curves;
- The '**zip()**' method for loops;
- Further operations with '**nuke.animation**' - working with keys and interpolations;

- The '**copyAnimation**' command to copy specific knob's animations between knobs;
- The use of '**fromScript()**' and '**toScript()**' methods to copy animations.

## Chapter 8:

- A method to set The framerange and views: '**nuke.getFramesAndViews('get range', '1-100')**'
- Introducing ***Functions***;
- Writing simple programs with functions;
- The use of the command '**setName**';
- Method for value's referencing inside a function – creating different error messages;
- The use of *global* variables – pros and cons;
- Using a function with the '**channels()**' method and '**nuke.display()**';
- The use of functions with animation;
- Syntax for setting specific incremental values on loops inside functions;
- Introducing '**simple panels objects**'**.**
- The use of a '**set**' to avoid duplicate objects in a list;
- Examples with simple panels objects and ideas for rewriting some previous program in a more efficient way;
- A method for picking up nodes and change the label and colors using a simple panel object;
- Example: Changing the filters for all Transform and Cornerpin Nodes from a simple panel Object;
- Example: the '**key stepper**' program (inspired by previous chapters);
- Example: the '**card creator**' (also improved from the previous chapter);
- The use of the '**addFilenameSearch()**' command inside a simple panel object to browse geometries;
- Example: A nice program to create a random 'Traffic' animation;
- Example: Creation of a '**comp optimizer**' or '**sanity check**' tool;

## Chapter 9:

- The "**while**" loop;
- Using the '**while**' loop to create multiple node's branches with specific nodes;
- The '**enumerate()**' method in a loop;
- Example: a program to create 3D geometries with the '**enumerate()**' method;
- Working out rotoshape and rotopaint nodes programmatically;
- The '**curvelib**' module;
- Managing rotoshapes parameters with '**getAttributes**';
- Methods for getting curve's vertices positions programmatically;
- Example: A nice program to stick small gradient balls on each curve's vertex of an animated curve;
- Methods to programmatically add and remove curve's vertices - the use of vector math;
- Creating shapes by using Python and the trigonometry;
- Methods for cloning shapes;
- A funny example: Randomizing curves' positions in the space;
- Dealing with Rotopaint brushes programmatically;
- Dealing with the roto's feathers;

## Chapter 10:

- Methods for Animation's curve manipulation;
- Animated curve editing: assigning keys, changing slopes and interpolations programmatically;

## Chapter 11:

- Creating patterns with expressions;
- Image manipulation with expressions;

## Chapter 12:

- TCL scripting;
- Integrating TCL in Python;
- TCL and node's automation;
- Creating a customized show's slate with TCL;
- Working with metadata;

## Chapter 13:

- Introducing Python's Dictionaries;
- Example: Creating a tool to store and manage selections of nodes with Dictionaries;
- Example: the use of dictionary for creating a tool to create and edit multiple shots arranged together in the timeline;
- The '**nukescripts.clear_selection_recursive()**' command;
- Example: rewriting the '**Comp optimizer**' tool with Dictionaries;
- Example: rewriting the '**shape creator**' program with Dictionaries;

## Chapter 14:

- Dealing with the '**nuke preferences**' programmatically;
- The '**menu.py**' and '**init.py**' startup files;
- Introducing the first Nuke's callback: '**nuke.addOnScriptLoad()**' and methods to set it inside the **init.py** file;
- The '**nuke.exists()**' command ;
- Creating special automatic functions with '**nuke.addOnScriptLoad()**' ;
- Further Nuke's callbacks: '**nuke.addOnUserCreate()**', '**nuke.addOnCreate()**';
- Python and the system's clock ;
- The '**nuke.addAutoLabel()**' command ;
- The '**nuke.addOnDestroy()**' callback !
- The most powerful Nuke's callback: '**nuke.addKnobChanged()**';
- Example programs with '**addKnobChange()**'**:** a program to change the node's colors based on its input status;
- The '**shown()**' method;
- The '**nuke.addBeforeRender()**' callback: **–** a smart program to check bbox sizes before rendering;
- The '**addBeforeFrameRender()**' callback - a nice program to check for rendering issues on each frame and print a final report;
- Operations with the Nuke's Viewer - '**nuke.activeViewer**' ;
- Image color sampling with the '**sample()**' method ;

- The '**nuke.addAfterRender()**' callback – a useful program to check for any zero pixes values after a rendering is complete;
- Writing a tool to check on rendering time on completion and for each frame;
- The '**addOnscriptsave()**' and '**addOnScriptClose()**' callbacks - examples;
- Setting Nuke's default values inside the **init.py** file: formats and knobs;
- The '**nuke.pluginAddPath()**' to set plugin paths inside the **init.py** file;
- Other techniques for installing and setting up gizmo and icons;
- Methods for importing template scripts;
- A smart method to Install 'hidden' tools;
- A sample program for creating a shortcut to toggle/untoggle the node's metadata display using Nuke's callbacks;
- Using callbacks to create a tool to disable an entire nodes' branch if a Merge node is disabled;
- The '**nuke.selectOnly()**' method;
- A smart tool: the '*good boy viewer*' program to manage the position of your Viewer in the DAG;
- Techniques for adding commands and setting preferences to Nuke by creating a special toolset;
- The Viewer: how to customise the knobs and the frame control;
- The '**Autohandle**' program to set your handles on the Viewers' timeline;
- Creating a progress bar to display the status of demanding processes;
- Example program: how to run a check on the 'white' knobs of all Grade nodes of a script and display a report;
- The Viewer node: how to create custom guides and mask on the Viewer;
- Setting the IP viewer process;
- Creating a program to check on frame's black edges on a rendered sequence;
- Creating a program to check on NAN pixels, negative values, edge stretches.

## Chapter 15:

- Advanced operations on Gizmo;
- Methods for adding knobs on a gizmo programmatically;
- Appending further knobs inside pre-existent menus;
- The use of '**knobChanged()**' in your Nuke sessions;
- Methods to create automations by using '**knobChanged()**' as one-line-long command program or with the triple quotes layout;
- The '**Command Menu**' knob;
- Creating a tool to switch between different Matrix effects using the '**Command Menu**';
- Method to disable/enable or hide/unhide knobs: '**setVisible()**' and '**setEnabled()**';
- Executing Nuke's menu commands;
- An interesting program: the '**goobBlacks**' tools to work out good black's level in the ACES2065-1 colorspace;
- Further nuke's methods:

    - **nuke.maxInputs(),**
    - **nuke.performanceInfo()**
    - **nuke.resetKnobsToDefault()**
    - **setName()**
    - **nuke.center()**
    - **nuke.createToolset()**
    - **nuke.createSceneFileBrowser()**
    - **nuke.executeMultiple()**
    - **nuke.fileDependencies()**
    - **nuke.askWithCancel()**
    - **nuke.choice()**

- **nuke.knobTooltip()**
- **nuke.recentFile()**

- - A smart and useful example: building a tool to store Viewer's connections to specific nodes and a method to call it back;
- - the '**nuke.connectViewer**' method;
- - the '**nuke.autoplace_snap_selected()**' command to align your nodes in the DAG view;
- - the **nuke.tabClose()** command;

## Chapter 16:

- Python CLASSES and examples;
- Creating Python panels;
- Creating multiple nodes using Classes and Python panels;
- Creating multiple 3D objects programmatically;
- Different methods to create nodes with Python panels: creating and arranging Cameras;

## Chapter 17:

- OS functions and path manipulation;
- Creating a tool to set the Environment path to a specific shot;
- The '**getFileNameList()'** method;
- The creation of a smart tool to import CG AOVs for a specific shot using a simple panel object: The **"Asset loader"**;
- Further methods for path manipulation:
  **'os.path.dirname()',**
  **'os.path.basename()',**
  **'os.path.isdir',**
  **'os.makedirs'**
- Creating a smart tool to 'version up' your comp scripts;
- Creating a quick but rather efficient 'path replacer';
- The '**nuke.scriptSaveAs()**' command;
- The '**endswith()**' method;
- The '**os.path.exists**' method;
- The '**nuke.filename()**' method;

## Chapter 18:

- Advanced Python Panel creation;
- Intelligent Python panel auto-creation routine based on the class of selected nodes;
- Creation of an advanced Camera manipulation tool with Python Panels;
- Examples with Python Panels and interactivity;
- A nice example with Python panels: The '**Card generator**';
- Building an efficient Camera calculator for Nuke;
- Application of the depth field equation to create a defocus calculator.

## Chapter 19:

- Using OS functions and methods to execute background processes;
- Commands to open, close and clear your scripts: '**nuke.scriptOpen**' , '**nuke.scriptClear()** ', '**nuke.scriptClose()**' , '**quit()**';
- Running Nuke from the command line;
- A simple program to create a '**Contact Sheet maker**';
- Reading and writing text files from Nuke;
- The '**isdigit**' method;
- Creating and publishing 'neutral grade' nodes using text files;
- Example : How to build a smart tool to compare between two Nuke's scripts by using dictionaries and OS functions;
- A further example with text files: creating the '**Telesnippet**' tool to share a snippet with other colleagues;
- The '**re**' module and methods for pattern matching;
- The '**glob**' module;

## Chapter 20:

- Introduction to pipelines;
- Planning your own comp pipeline;
- Creating the '**Show Saver**' for loading and saving your comp scripts in your pipeline;
- Creating the '**Asset Manager**' to import assets from all departments;
- Implementing an '**autocomp**' feature on the Asset Manager;
- Creating your custom Write node for the pipeline;
- An easy way to 'publish' a shot ;
- Useful Python scripts to embed inside the **init.py** file to improve pipeline capabilities;
- The '**nuke.scriptSaveToTemp**' method for creating safe backup copies of comp scripts;
- conclusions.