Sound Processing  
 Sound Editing  
 Lua

Ryan Oo

Jai Raina

Andrew Wong

Hardware / software presentation

Software Design & Development 2016

# Hardware Category: Sound Processing

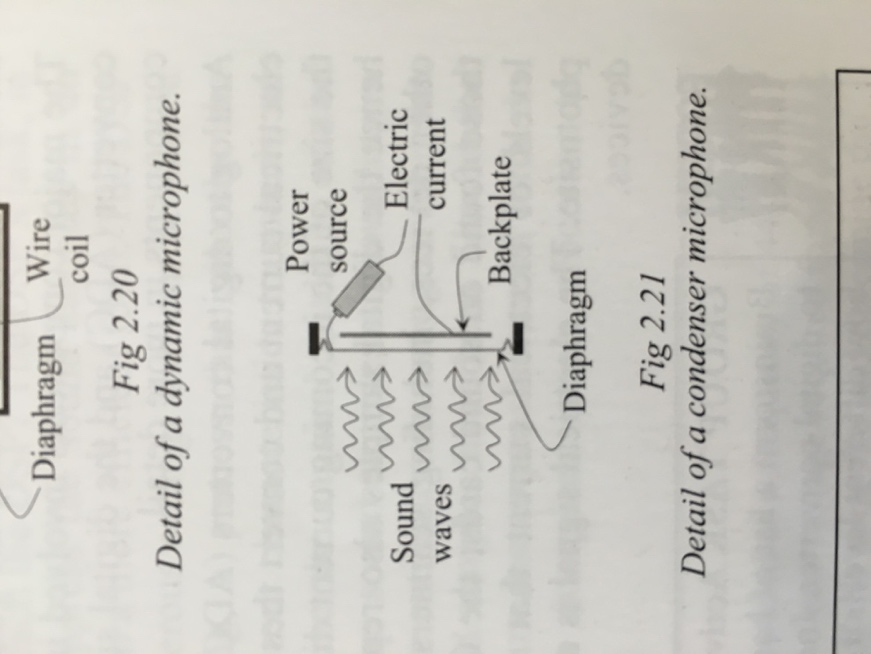
## **Overview**

### Analogue and Digital Waves

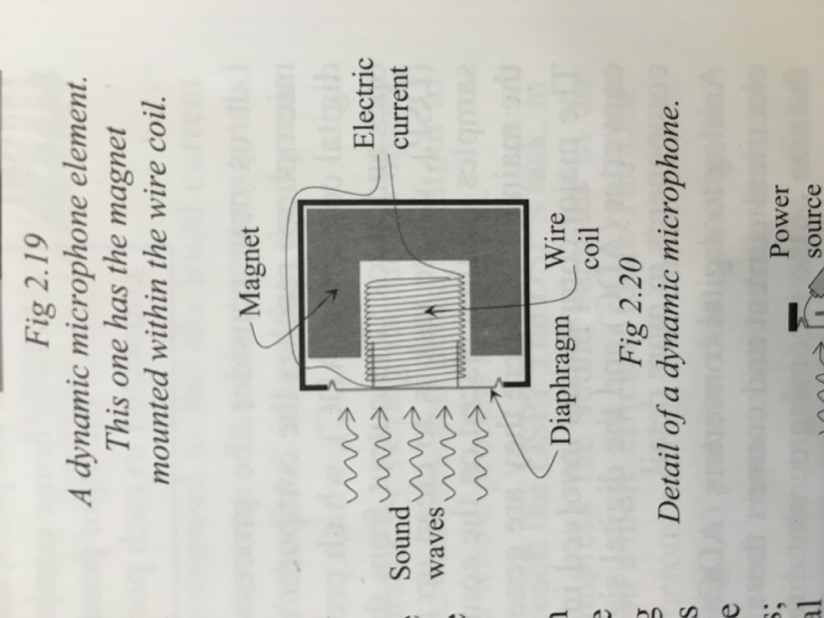
The sound that we hear is analogue sound. It is a series of compressions and rarefactions in a medium, in this case air. Our ear receives the compression and rarefactions and sends electrical signals to our brain which interprets it as sound. However, a method of capturing sound needs to be made for computers, as they use binary. To do this, we have to transfer analogue waves to digital waves. A microphone does this. There are many types of microphones, but the two most common ones are condenser and dynamic microphones.   
  


### Types of Microphones

**Condenser** microphones have a diaphragm, which is considered the front plate of the microphone. The diaphragm vibrates in and out in response to the compressions and rarefactions of the soundwave. A back plate behind the diaphragm stays still. An electric current flows between the diaphragm and back plate. As the diaphragm moves closer to the back plate in response to the soundwave, the electrical current flows more freely to the back plate. As the diaphragm moves further away from the back plate, the current flow is reduced. Thus, the electric current represents the changes in the sound wave. The electric current is an analogue signal of the sound wave, and is sent to an ADC (analogue to digital converter), which transforms the analogue signal into digital data (binary).



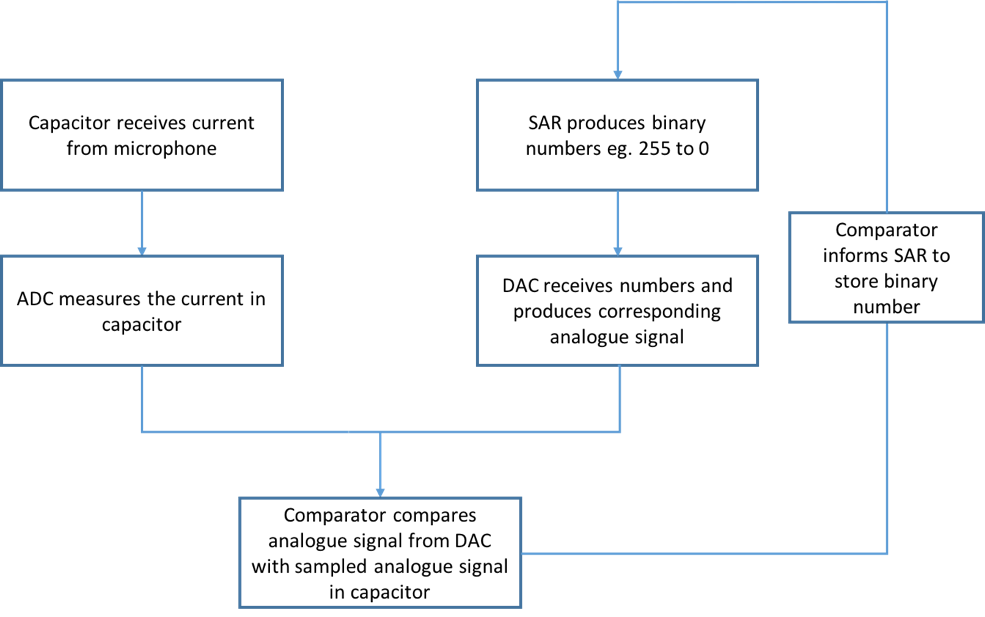
**Dynamic** microphones have three important components, a diaphragm, a copper coil of wire and a magnet. The diaphragm is attached to the copper coil of wire, and the copper coil of wire is surrounded by the magnet. As the diaphragm moves in and out in response to the sound wave, so does the copper wire. This causes the copper wire to create an electric current with the magnetic field. The analogue signal (electrical current) in the copper wire is sent to an ADC, where it transforms the electrical current into digital data.



## **How It Works**

### Sound Sampling

There are five devices that allow sound sampling to take place; ADCs (analogue to digital converters), DACs (digital to analogue converters), SARs (successive approximation register), DSPs (digital signal processer) and comparators. The analogue signal from the microphone is fed into a capacitor at precise intervals eg. 44100Hz. The ADC measures the current of the analogue signal in the capacitor. A successive approximation register (SAR) produces binary numbers eg. 255 all the way down to 0. The DAC receives these numbers and produces the corresponding analogue signal. A device called a comparator compares the analogue signal from the DAC with the sampled analogue signal in the capacitor, if it is the same analogue signal, the comparator informs the SAR, and the SAR stores the number generated.



This number represents a sample of the ***amplitude*** of the analogue wave captured by the microphone and is converted into a series of ones and zeroes to represent binary data. This process is repeated multiple times a second, eg. at the rate of 22,000Hz and 44,100Hz, which is the speed of sampling achieved with modern technology. An audio file sampled at a rate of 22,000Hz will have less quality than an audio file sampled at 44,100Hz, but it would be half the file size of the 44,100Hz sound sample. The sound samples are then sent to a DSP (digital signal processor), which can perform a variety of tasks on the sampled sound, including wave shaping, to make the transitions from sample to sample smoother. DSPs can also remove background noise from music, as background noise has different characteristics than music.

To reproduce the sampled sound, the amplitude values captured by the sound sampling is converted into an electrical current. High amplitude = High current, low amplitude = low current. The speaker contains a diaphragm which is connected to a copper coil of wire. The copper coil of wire is surrounded by a magnet. When the electrical current is applied to the copper coil, it causes the magnetic field to fluctuate, vibrating the copper coil in response to the changing of the magnetic fields. This causes the diaphragm to also move, causing compressions and rarefactions in the air we hear as sound.



## **Historical Development**

### Acoustic Era (1877 to 1925)

The quality of sound captured has vastly improved over the years, as has many aspects of technology. The earliest form of recording sound was called the acoustic era. A phonograph was used to record sound during this era and it had a large horn that collected the sound waves around it. A diaphragm was connected to the apex of the horn, that moved according to the compressions and rarefactions in the sound wave. The diaphragm was connected to a scribe, and as the diaphragm moved in response to the soundwaves picked up by the horn, the scribe scratched the surface of a soft metal to record an analogue of the soundwave on the surface of the soft metal. This method of recording sound was only able to capture sounds of frequencies between 250Hz and 2,500Hz and only allowed a low quality of sound to be recorded.

### Electrical Era (1925 to 1945)

This was the second era of sound recording, and employed the use of electronic microphones to collect the sound, electronic amplifiers to strengthen the sound and electronic disc cutters to record the sound on metal discs. Condenser and dynamic microphones were used to transform analogue sound into analogue electrical signals. These signals were fed to an electronic amplifier which strengthened the signal, making the analogue sound louder. The signal then went to an electronic disc cutter, which transformed the electrical signals into mechanical movement, scribing the analogue onto a polyvinyl plastic disc. This new improvement in sound recording allowed frequencies of 60Hz to 6,000Hz to be recorded, and improved sound fidelity.

### Magnetic Era (1945 to 1975)

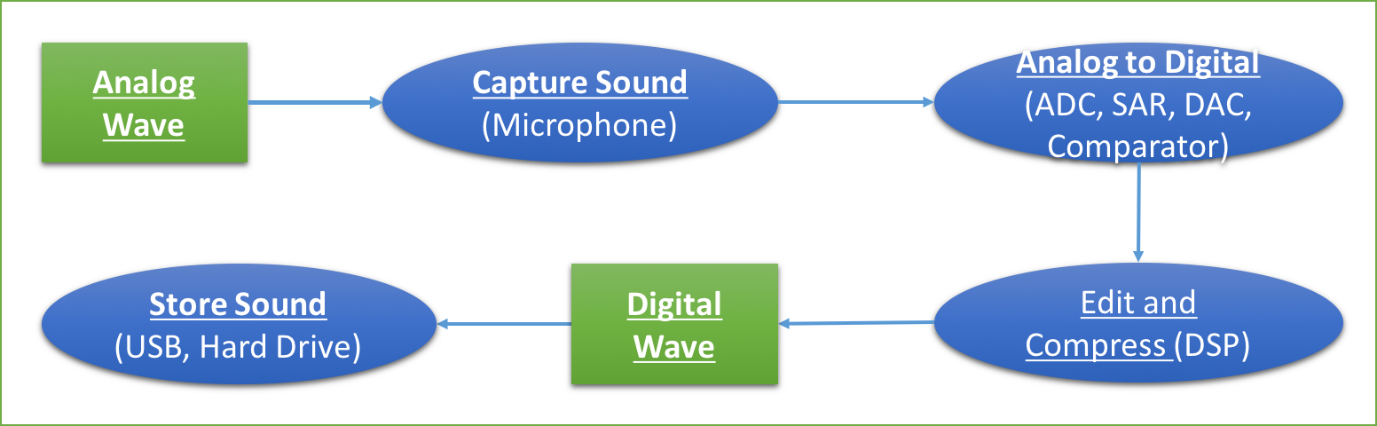
The third era of sound recording involved the use of magnetic tape recorders. A strip of magnetic material moves at a constant speed past an electronic write head. An analogue electrical signal captured by a microphone is fed to the write head. The write head passes the electrical signal to the magnetic strip running past the head, inducing a pattern of magnetisation similar to the signal in the write head. This method of storing sounds greatly increased the fidelity of the sound recorded, and also allowed the storage medium to be more portable, as the method of storage is more efficient.



### Digital Era (1975 to now)

The fourth and current era of sound recording captures a discrete signal many thousands of time a second eg. 44,100Hz. An analogue signal from a microphone is transferred into binary data by a variety of means (indicated previously in the seminar notes), and can be stored on a variety of devices eg. USBs, Hard Drives and CDs. USBs use a method of electrically storing recorded sound, Hard Drives use a magnetic method of storing recorded sound and CDs use an optical method of storing sound. Digitally stored sound are able to capture sound of the highest and most accurate fidelity and are able to produce frequencies 20Hz to 20,000Hz (the sound range that humans are able to hear). Digitally stored sound is even capable of storing sound in more frequencies than humans can hear, but it would not be practical. Along with this, digitally stored sound is extremely portable, as it can be carried in your pocket, in the case of USBs and can fit in your hand, in the case of CDs.

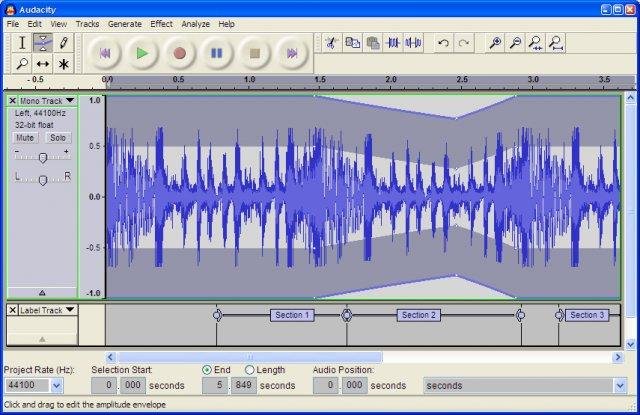
## **Journey of Analogue Sound to Digital Storage**

****

# Software Category: Sound Editing

## **Overview**

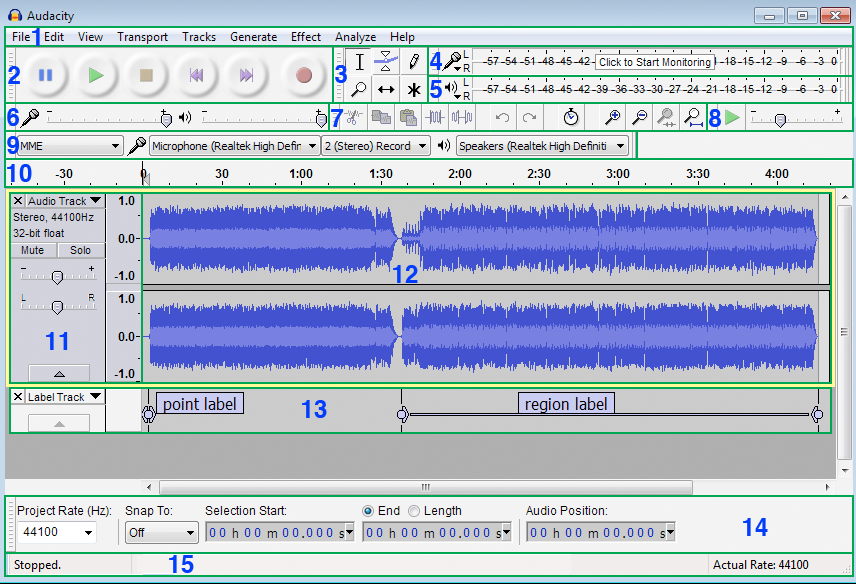
Sound editing has been present in all of society for decades; mainly through the entertainment industry. Movies, television, games and of course music require sound editing. Not only do musicians or filmmakers necessitate over sound editing; but students have potential use for it as well for school assignments, projects or for recreational leisure. Sound editing has been achieved in the past and the present through 2 key ways. Analogue editing and digital editing. Analogue editing is a traditional tape-based post-production and in contrast to digital editing which uses software based tools on a computer. A few recognisable examples of digital sound editing software include: Audacity, Wavepad, FL Studio, Adobe Audition, just to name a few. For the rest of the ‘Sound Editing’ portion of this seminar, the presentation will be focused and based on the sound editing software known as Audacity.





## **How It Works**

Audacity is a multi-track audio editor and recorder which includes a variety of features which help users in creating and editing audio files. Below and to the right is a guide on how to use and navigate Audacity.



1. **Menu Bar –** Offers commands such as: file, edit, view, transport, tracks, generate, effect, analyse, and help.
2. **Transport Toolbar** – Provides buttons for controlling playback, recording and moving the project to the start or the end; all of which are achieved by the pause, play, stop, skip to start, skip to end and record buttons.
3. **Tools Toolbar –** Provides buttons known as:

* Selection (which can be used to mark a starting point for audio playback);
* Envelope (allows volume changes)
* Draw (which can also be used to change volume),
* Zoom (which can be used to zoom in and out of the audio track to see greater detail of certain sections)
* Time shift (allows to synchronize audio in a project by dragging individual or multiple audio tracks, note tracks or audio clips left or right along the ‘Timeline’)
* Multi tool (allows access to all of the five separate tools on the toolbar at once.)

1. **Recording Meter Toolbar –** Displays audio being recorded
2. **Playback Meter Toolbar –** Displays audio being played
3. **Mixer Toolbar –** Changes recording volume (on the left) or the playback volume (on the right)
4. **Edit Toolbar -**  Provides buttons such as: cut, copy, paste, trim audio, silence audio, undo, redo, zoom in, zoom out, zoom selection, and fit project; all of which are self-explanatory. However the Sync-Lock Tracks button requires a little explaining. The button ensures that length changes occurring anywhere in a defined group of tracks also take place in all audio or label tracks in that group, even if those tracks were not selected. This lets you keep existing audio synchronized with each other.
5. **Transcription Toolbar –** Allows users to play audio at slower or faster speeds than normal.
6. **Device Toolbar –** Helps in selecting: audio host, recording device, recording channels and playback device
7. **Timeline –** Displays a horizontal ruler above the tracks measuring time from zero (the start of the track)
8. **Track Control Panel –** It is actually a part of the audio track which Includes controls to change the track. Controls include:

* Close Button – Closes the track, removing it from the user’s project
* If the user clicks the Audio Track Dropdown Menu, more controls are provided such as the:
* Mute button which silences the track selected
* Solo button which plays the single track selected
* Gain slider which sets the gain for track selected
* Pan Slider which makes signal stronger on left or right earphone

There are also status indicators which provides information on the track: showing if it is stereo or mono; the sample rate (Hz); and the sample format.

1. **Audio Track –** Shows the users a Vertical Scale with units. This scale displays amplitude when showing the waveform, or frequency when showing the spectrum or pitch. The audio track also provides a visual representation of the audio waveform. Tracks like these are created whenever a new audio file is recorded or imported.
2. **Label Track –** This is an additional track which can be added into a user’s project. Labels are used reference points or regions in an audio track. However the label track itself does not holdaudio. In addition, labels can also be used to contain text to annotate a certain part of the track.
3. **Selection Toolbar –** This includes controls for ‘Project Rate’ (sample rate), ‘Snap To’ and for manipulating units of time or other selection formats.
4. **Status Bar –** This bar is always at the bottom of the page, and its core function is to display messages about recording or playback; indicate parts of Audacity as well as provide hints on how to use Audacity for the user.

## **Main Editing Features of Audacity**

Audacity provides great features for users to work with. Examples of these great features include:

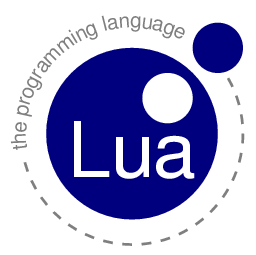
* Simple manipulation and editing using the cut, copy and paste buttons
* Unlimited sequential undo and redo, giving users more flexibility; encouraging them to try new things.
* Mixing and compiling as many tracks as users need which allows freedom for creativity. This technique can be completed using 2 ways. The first procedure is called ‘Mix and Render’ and the second one is called ‘Mix and Render to New Track’. Mix and Render simply mixes down any selected tracks; to a single mono or stereo track, rendering to the waveform all real-time transformations that had been applied. The resulting track also known as the ‘Mix’ track replaces the selected tracks and is placed underneath any tracks that were not mixed and rendered. The other technique, ‘Mix and Render to New Track’ is almost the same thing. The one major difference being the original tracks are preserved rather than being replaced by the ‘Mix’ track.
* Putting in unique sounding effects or in contrast reducing sound; depending on what the user is trying to achieve. This can be accomplished by clicking the effect button, where the user can do a variety of things, for example: change the tempo, change the pitch, reduce noise, add in an echoing effect or amplify the audio.

## **Historical Development**

Audacity, the audio editing software was first worked on by Dominic Mazzoni and Roger Dannenberg in 1999, as a research project for university. It was first released on the 28th of May, 2000 as version 0.8. By June, 2002 they released version Audacity 1.0. Near the end of 2003, Audacity released the 1.2 series, which introduced hundreds of new features such as; support for professional-quality 24-bit and 32-bit audio, a comprehensive manual, and it provided non-English speaking languages supporting other countries. After the run of the 1.2 series, a Beta 1.3 series with enhanced features was released which serves as a foundation for the current 2.0 series for Audacity. This Beta that was released had new features such as: new keyboard commands, new shortcuts, the ‘Snap To’ button was introduced, an enhanced interface, new effects, and overall improvements towards the whole software. As more versions were slowly being released, incremental changes were being applied to Audacity. With each new version came more bug fixes, improved interfaces, new effects, new files could be imported and exported. By version 2.0, Audacity was highly enhanced and improved from its predecessors. During the 2.0 series (which is still running today) creativity and freedom for users is one of the main goals for Audacity. Giving users expansive boundaries to go towards. Audacity is now smoother, more effective and has more variety for users to work with. Audacity is now available for Windows, Mac, OS X, Linux, and offers over 30 languages for users.

# Programming Language: Lua

## **Overview**



Lua is a cross-platform, extension programming language that appeared 23 years ago in 1993 – publically in 1994. It was created to serve as a lightweight scripting language that could extend and customise the functionality of programs. In other words, the Lua programming language was not the primary programming language of a program, but rather a side language, which helped increase development efficiency.

Lua was developed in the Pontifical Catholic University, by three people: Roberto Ierusalimschy *(Ee êh roo zah leems key)*, Luiz Henrique de Figueiredo, and Waldemar Celes. During 1977 till 1992, Brazil has a strict market reserve (aka trade barrier) that made it difficult for them, as members of their university’s Computer Graphics Technology Group (Tecgraf) to buy nor sell custom computer software. So, in response? They built their own software.  
  
Lua is a programming language very similar to other languages, and should be easy to pick up on. It has the dynamic scripting feel like JavaScript, where the variables are not assigned a type, but rather the values only - and also has a similar code syntax with C and Pascal, two other programming languages.

Fun fact: the name of this programming language, Lua, is the Portuguese word for Moon. This programming language was built as a successor of a previous language, called SOL (Simple Object Language). As SOL meant Sun in Portuguese, the creators of Lua decided to name their programming language the opposite of Sol, as a joke.  
  
Fun fact #2: the programming language was awarded the 2011 Game Developer Magazine Front Line Award

## **How It Works**

Creating Lua Scripts

Lua scripts are simply plain-text files, and hence can be written with any text editor. However, special editors, known as Integrated Development Environments (IDEs) have features that assist the developer, such as code completion, syntax highlighting, error checking, and etcetera.  
  
Lua scripts can be identified if their file extension are “.lua”

### Programming Syntax

Like many other programming languages, Lua is not strict on the formatting of the code itself – spaces, line breaks and indentations do not affect the operation of the software. Dreaded by some, the semicolon, used to indicate the end of a line, is optional.

At the basic level of Lua, it is also able to do mathematical calculations and comparisons. Their syntax is shown below  
  
+ add 1 **+** 2 3

- subtract 9 **–** 4 5

\* multiply 2 **\*** 3 6

/ divide 9 **/** 9 1

% modulo (remainder after division) 9 **/** 6 3

^ power 2 **^** 2 4

== is equal to 9 **==** 2 false

~= is not equal to 9 **~=** 2 true

< is less than 9 **<** 2 false

> is greater than 9 **>** 2 true

<= is less than or equal to 5 **<=** 6 true

>= is greater than or equal to 5 **>=** 5 true

# length of **#**{1,2,3} 3

-- comment (until end of line) ... **--**comment  
--[[ open comment block ... **–-[[** comment block  
--]] close comment block spam spam spam **--]]**

0.5 number

1 number  
“string” string

‘string’ string

\ escape character

nil undefined / null

### Syntax/Language Comparison

* End of line semicolons are optional
* Braces {} are replaced with do and end.
* Arrays are 1-based (array position starts from 1)  
   print(({"A","B","C"})[0]) nil  
   print(({"A","B","C"})[1]) A
* No ternary operator (a?b:c) – aka no ‘one liners’
* Multiple assignment x,y = y,x
* Numbers are numbers (No difference between integers and floats)
* Undefined/null values are represented as nil

### Variables

#### Variable Types

Variables are data items which store information. In Lua, also known as an identifier, a variable can take form as a global variable or a local variable (or a variable in a table – a property)  
A global variable is accessible at any point of the code. It can be declared by variable = 10  
A local variable can only be accessed by the functions in the same nesting, and can be declared by appending the keyword local before the variable declaration. local var = 3

If a variable is referenced without being declared, it will be undefined, or nil

#### Variable Naming Conventions

Variables can be named anything, provided that they:

* Are not the same as Lua’s programming keywords
* Contain only alphanumeric and underscore characters (A-Z, a-z, 0-9, \_)
* Do not begin with a digit

### Memory Management

In context of software development, programmers need to be aware of how much memory their software is consuming. If their code is laden with too many unused variables, or has a memory bug, the application may freeze and crash, or even worse, crash the whole computer. To stop these memory leaks, the programmer must be aware of how much memory their application is using, and take necessary memory management precautions  
  
The Lua programming language has inbuilt automatic memory management, also known as Garbage Collection (GC). At regular intervals, the program will automatically free up memory by releasing variables that will no longer be used. Garbage Collection, however, does not affect global variables, nor properties in a table.

To delete variables, simply set the variable to a value of nil

## **Advantages Over Other Languages**

* Easier use for prototyping compared to other languages
* Works on many platforms – works from microcontrollers to game engines
* Shorter development time (compared to C++ for example)
* Compilation can occur at runtime (code does not have to be manually compiled)
* Lightweight (not too much memory overhead)
* Relatively simple syntax – very close to pseudocode

## **Historical Development**

Lua has undergone many updates, and is currently at version 5.3.2, which was released in November 2015. Each iteration brought new features, functions, bug fixes (and bugs…), more efficient code execution, and support to more and more devices.

## **Applications**

As mentioned earlier, Lua is not intended as the primary programming language of an application (albeit possible). Instead, Lua is embedded in a program with another main language. It is commonly used in games for scripts that are triggered in a level.  
  
Well known software/games/whatever that use Lua include: the VLC player, TeamSpeak, Garry’s Mod, CryEngine, Adobe Lightroom, World of Warcraft, and many more. It also can be used to script the LEGO Mindstorms NXT unit.

# Bibliography

Software Design and Development Preliminary Course – Samuel Davis. Page 61

<https://en.wikipedia.org/wiki/Sampling_(signal_processing)>

<http://www.tech-faq.com/how-does-usb-work.html>

<https://en.wikipedia.org/wiki/History_of_sound_recording>

<http://ethw.org/Electrical_Recording>

<http://www.lloydmicrophoneclassics.com/mic_history.html>

<http://computer.howstuffworks.com/flash-memory1.htm>

<http://www.g-w.com/pdf/sampchap/9781590707678_ch24.pdf>

<http://wiki.audacityteam.org/wiki/About_Audacity#Short_history>

<http://www.audacityteam.org/about/features/>

<http://manual.audacityteam.org/o/index.html>

<https://en.wikipedia.org/wiki/Audacity_(audio_editor)>

<http://www.afterdawn.com/software/version_history.cfm/audacity>

<http://manual.audacityteam.org/o/man/mixing.html>

<https://docs.coronalabs.com/daily/guide/start/introLua/index.html#memory-allocation>

<https://en.wikipedia.org/wiki/Lua_(programming_language)>

http://www.lua.org/uses.html