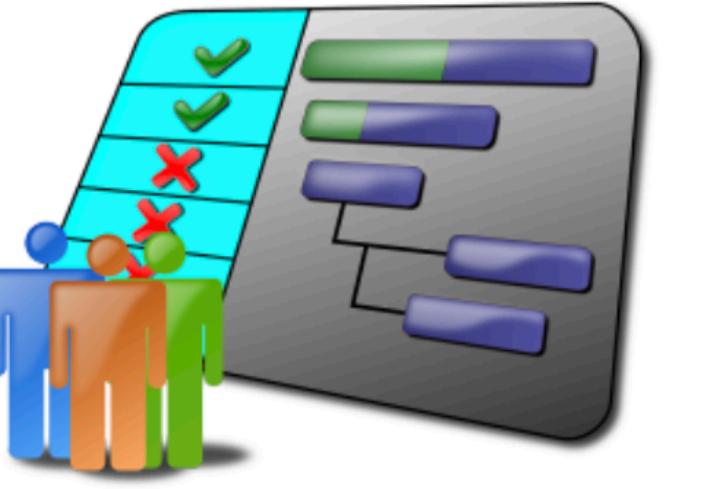
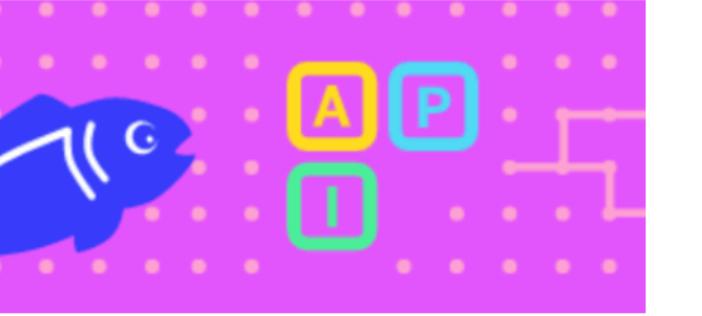
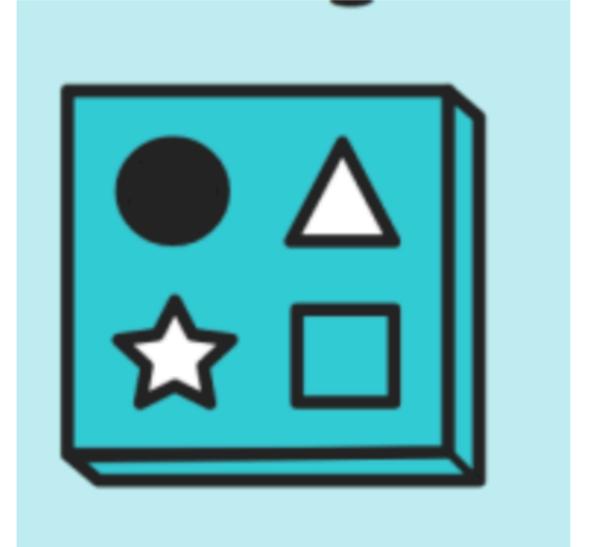
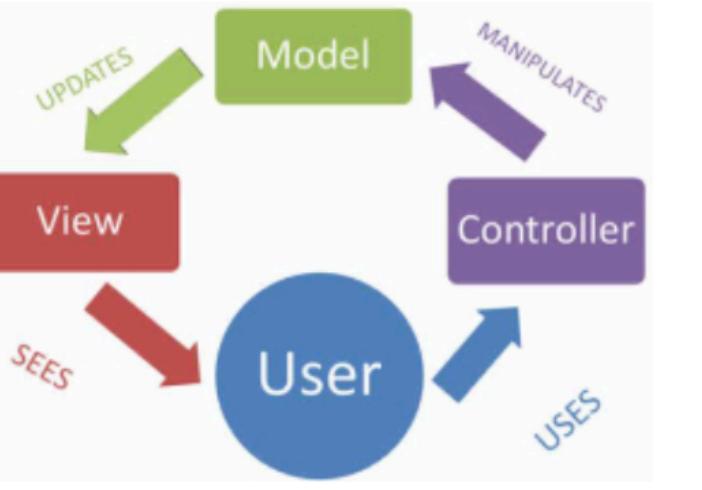
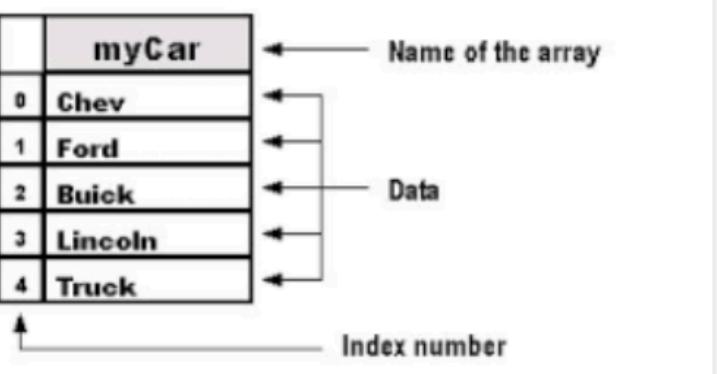
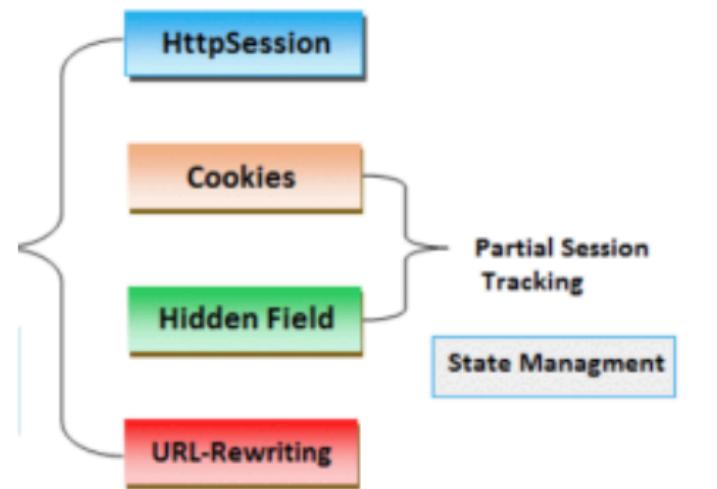


ICT Skills Module Overview

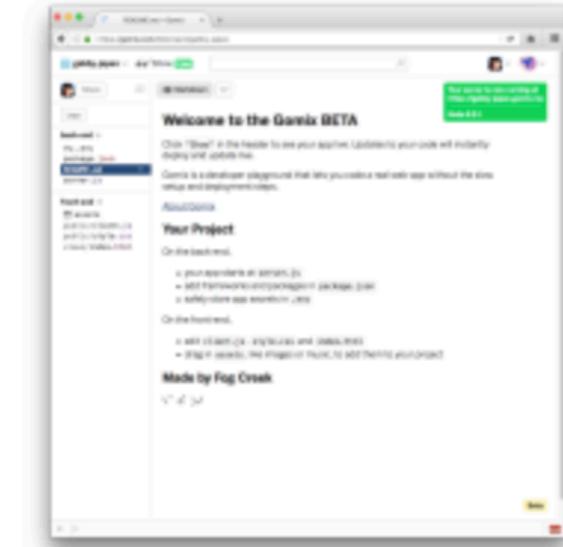
Module Topics

0: Overview & Assignment  	1: Introduction to Glitch   <p>An introduction to the Glitch platform + the very basics of the Javascript Language</p>	2: Controllers & Views   <p>Build your first Glitch app, a simple static playlist web site.</p>	3: Templates & Routes   <p>Template engine</p> <p>Explore templating in more detail. Enhanced the routing behaviour</p>
---	---	--	--

4: Model View Controller  <p>Explore MVC as implemented in Playlist</p>	5: JS: Arrays  <p>Comparison of an array to a column of data</p> <table border="1"><tr><td>myCar</td><td>Name of the array</td></tr><tr><td>0 Chev</td><td>Data</td></tr><tr><td>1 Ford</td><td></td></tr><tr><td>2 Buick</td><td></td></tr><tr><td>3 Lincoln</td><td></td></tr><tr><td>4 Truck</td><td>Index number</td></tr></table>	myCar	Name of the array	0 Chev	Data	1 Ford		2 Buick		3 Lincoln		4 Truck	Index number	6: Sessions  <p>In order to implement user account management, sessions provide a mechanism for identifying specific users</p>
myCar	Name of the array													
0 Chev	Data													
1 Ford														
2 Buick														
3 Lincoln														
4 Truck	Index number													

1: Introducing Glitch

Introducing Glitch



What is it, what role it plays, why was it built.

Glitch Tour



Building Blocks

A look at the components of a glitch project. Also types of project will we build?

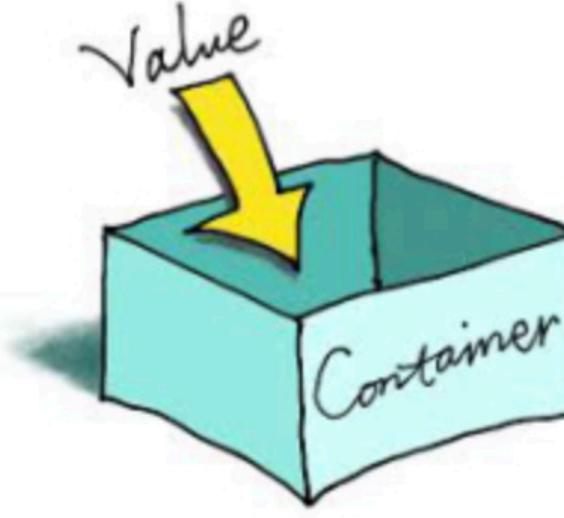
JS Introduction



JavaScript

Place javascript in its proper context, and explore its relationship to the browser.

Variables



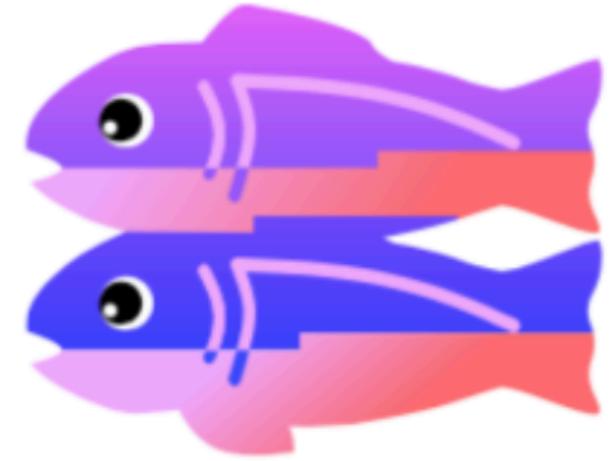
Explore the javascript variables, including the basic types, conversion and usage

Const, Let & Objects



Using const & let. Declaring and using objects.

Lab-1 Glitch Intro



Create, modify and view your first Gomix project.

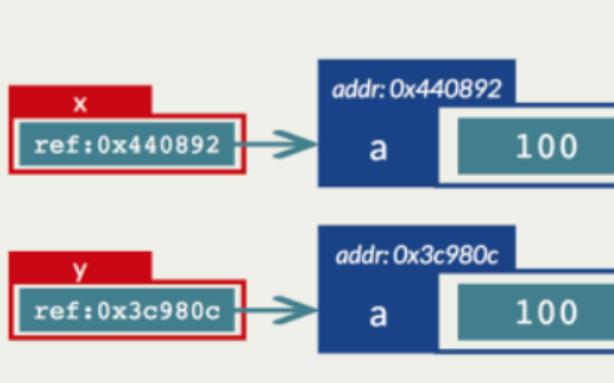
Lab-2 JS Intro



Background & Tools, Variables & Boolean Logic

2: Javascript Introduction

Variables & Objects Review



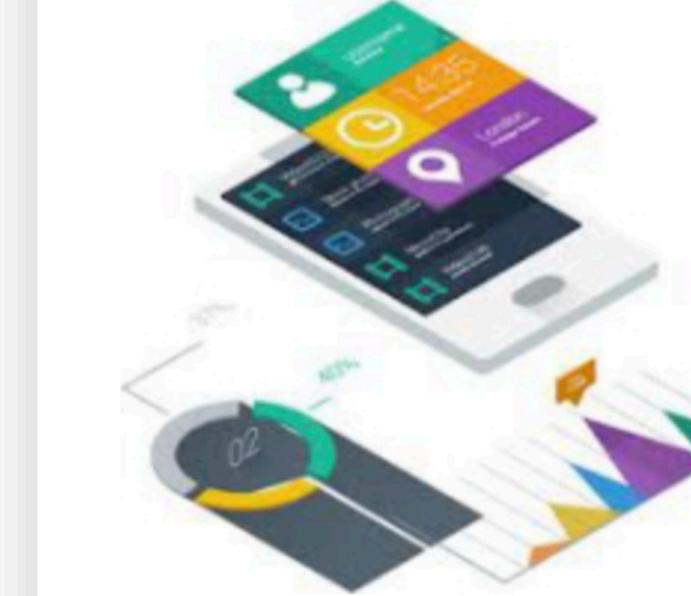
A concise tour of the structure of variables & objects in Javascript

Web App Introduction



Structure of a web app: Front-end Vs Backend. Routers, Models, Views, Controllers

Front-end



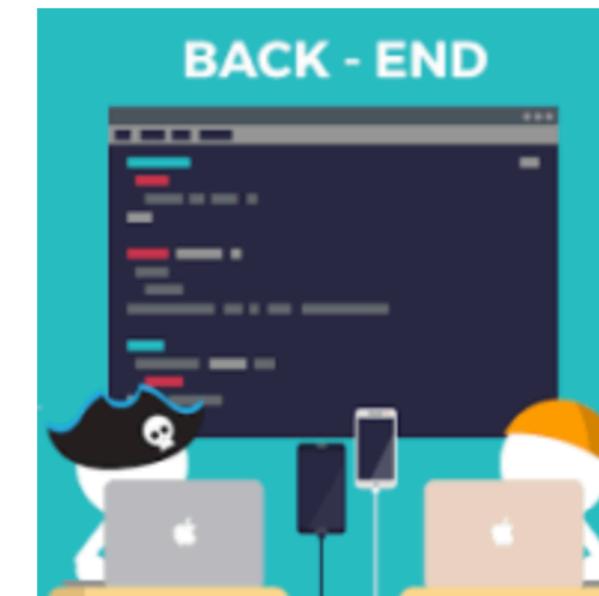
Views: Handlebars layouts, partials and templates

Modules



The backend will use a modular approach, relying on specific mechanism to import/export shared objects

Back-end



Server, routes + controllers

Lab-3 Playlist 1



Import and run a new starter project. Extend this project to include multiple 'views'. Explore the handlebars templating library.

3: Templates & Routes

Templates

Template engine



Templates enable dynamic composition of views from layouts, partials and expressions.

Json

```
"playlistCollection": [
  {
    "title": "Beethoven Sonatas",
    "songs": [
      {
        "title": "Piano Sonata No. 3",
        "artist": "Beethoven"
      },
      {
        "title": "Piano Sonata No. 7",
        "artist": "Beethoven"
      },
      {
        "title": "Piano Sonata No. 10",
        "artist": "Beethoven"
      }
    ]
}
```

JSON is notation for representing javascript objects in a simple literal format.

Dashboard

Beethoven Sonatas	
Song	Artist
Piano Sonata No. 3	Beethoven
Piano Sonata No. 7	Beethoven
Piano Sonata No. 10	Beethoven

Beethoven Concertos	
Song	Artist
Piano Concerto No. 0	Beethoven
Piano Concerto No. 4	Beethoven
Piano Concerto No. 6	Beethoven

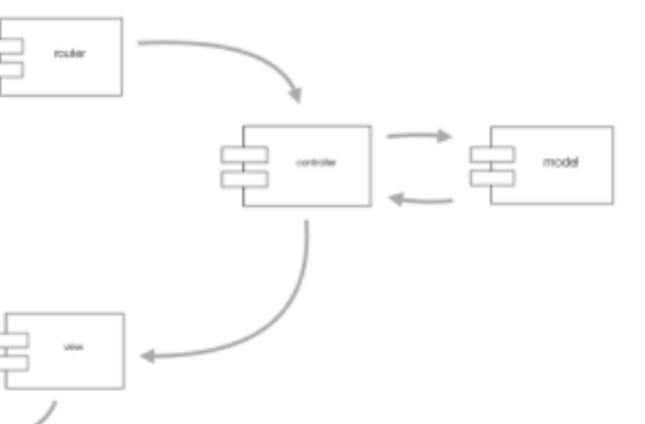
Review the dashboard controller in detail.

Playlist

Beethoven Sonatas	
Song	Artist
Piano Sonata No. 3	Beethoven
Piano Sonata No. 7	Beethoven
Piano Sonata No. 10	Beethoven

Revise the Dashboard to render playlist without their contents.
Use a new playlist view renders individual playlists

MVC



Explore the MVC Pattern in action in Playlist 2

Lab-4 Playlist 2



Refactor the dashboard controller to show summary on of the playlists + link to show playlist details.

4: Model View Controller

Module View Controller

The diagram illustrates the MVC (Model-View-Controller) pattern. It features a central blue circle labeled "User". Four arrows point towards it from the surrounding components: a green arrow labeled "UPDATES" points from "View" to "User"; a purple arrow labeled "MANIPULATES" points from "Controller" to "User"; a red arrow labeled "SEES" points from "View" to "User"; and a blue arrow labeled "USES" points from "User" to "Controller".

MVC is the guiding principle for the structure of our application.

Delete Song

Artist
Beethoven
Beethoven
Beethoven

How to remove a song from the playlist

Forms Design

How a form UI is laid out in HTML using Semantic UI

Form Programming

How to accept user input from a form and process it in a controller

The Store

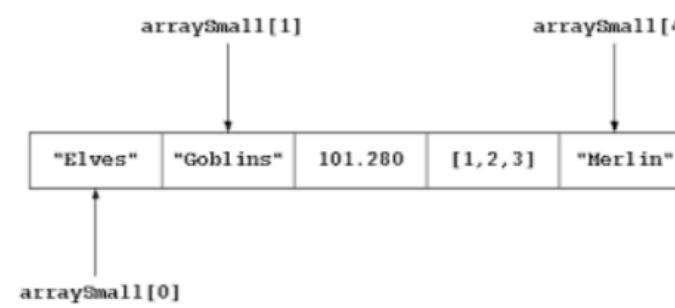
The Playlist are ultimately stored in a JSON file. This file is managed by database modules.

Lab-5 Playlist 3

Enable Songs and Playlists to be added via simple forms.

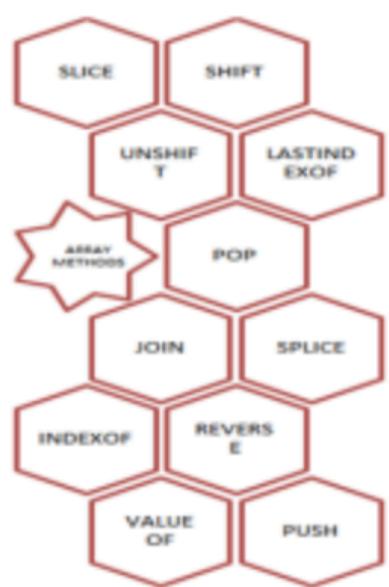
5: Javascript Arrays

Arrays: Basics



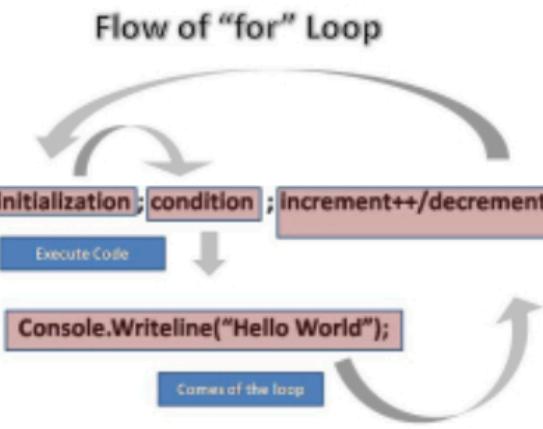
Creating, accessing, adding to and removing from arrays.

Array Methods



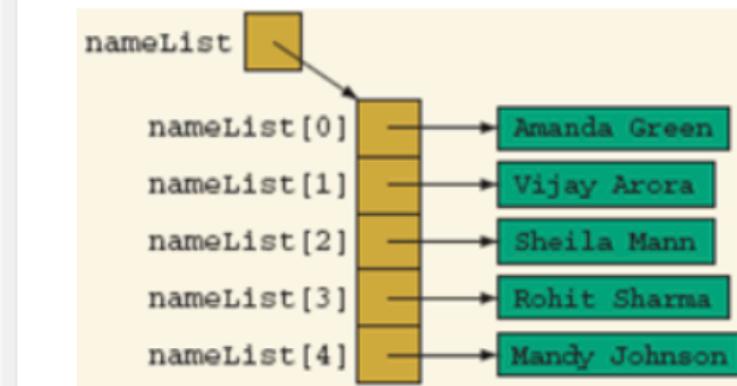
Exploring length, slice, concat, join, indexOf, lastIndexOf

Array Iteration



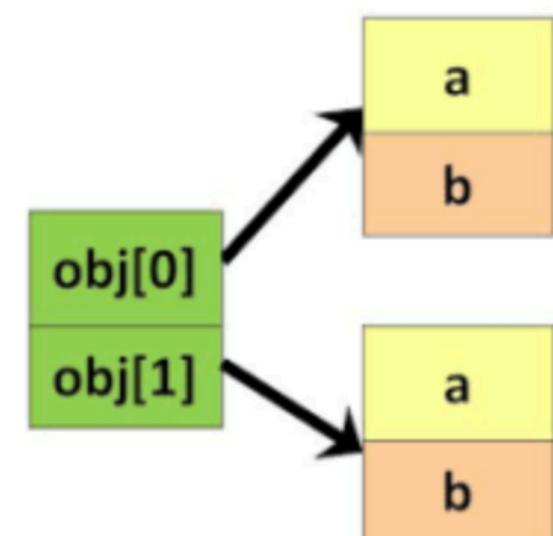
Using for, while and do-while to iterate over an array

Arrays of Strings



A review of the structure of arrays of Strings

Arrays of Objects



Arrays of more complex data structures, including nested objects.

Lab-6 JS Arrays

	myCar	Name of the array
0	Chev	Data
1	Ford	
2	Buick	
3	Lincoln	
4	Truck	

Comparison of an array to a column of data

Array Basics, Array Methods & Iteration

6: Sessions

Sessions Introduction

The diagram illustrates four methods for session tracking:

- HttpSession**: Represented by a blue box.
- Cookies**: Represented by an orange box.
- Hidden Field**: Represented by a green box.
- URL-Rewriting**: Represented by a red box.

A bracket groups **Cookies**, **Hidden Field**, and **URL-Rewriting** under the heading **Partial Session Tracking**. A bracket also groups **HttpSession** and **Cookies** under the heading **State Management**.

Keeping track of the currently logged in user is a challenge - as HTTP is, by definition 'stateless'. Hidden form fields, url rewriting and cookies are three common techniques for implementing sessions.

Using Sessions

The diagram shows the session flow between a **Web Browser** and a **Web Server**:

- A **Web Browser** sends a **GET /welcome.php?name=William** request to a **Web Server**.
- The **Web Server** responds with a **201 OK** status and sets a **Set-Cookie: PHPSESSID=12345** header.
- The **Web Server** stores the session state **name=William** in a **Session Store**.
- A second **Web Browser** sends a **GET /next.php** request to the **Web Server**, including the **Cookie: PHPSESSID=12345**.
- The **Web Server** retrieves the session state from the **Session Store** and responds with a **200 OK** status, displaying the content **<html><hi William...</html>**.

Sessions UX

Explore how we need to refactor the application to support sessions

New forms needed to enable the user to signup / login

Creating Sessions

The diagram compares two session management approaches:

- Memory using Cookies** (client side): Stores session data in the browser's memory.
- Session Cookies** (server side): Stores session data on the server.

Session Cookies are highlighted with a circle and include the following features:

- Used even when cookies disabled
- Force explicit login
- Deleted when browser closed
- Can only be read by site that created
- Distinguishes Guests
- Identify user by computer they used

Lab-7 Playlist 4

Playlist 4

Log-in

Email: homer@simpson.com

Password: *****

Login

Introduce Sessions onto the Playlist application, enabling user accounts and cookie-based authentication.