SYST24444

Mobile Web-Based App Development

Agenda

- Mobile Ecosystem
- JavaScript Review

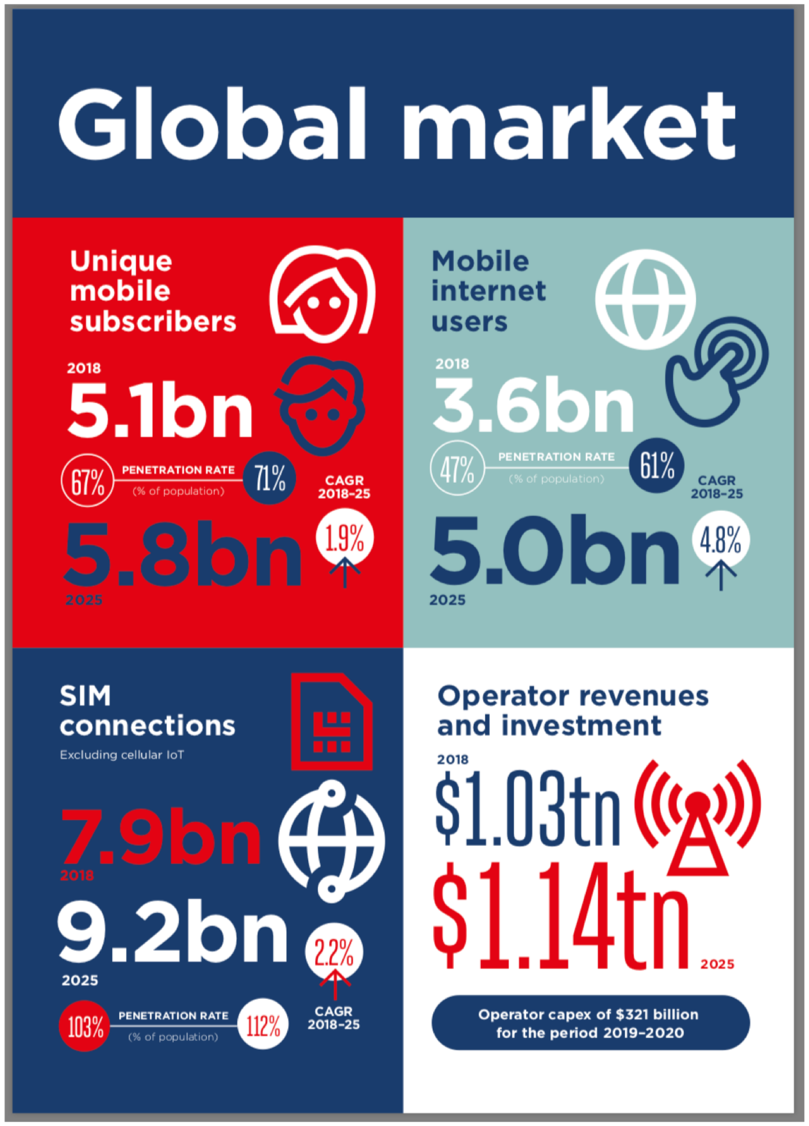
The Mobile Economy

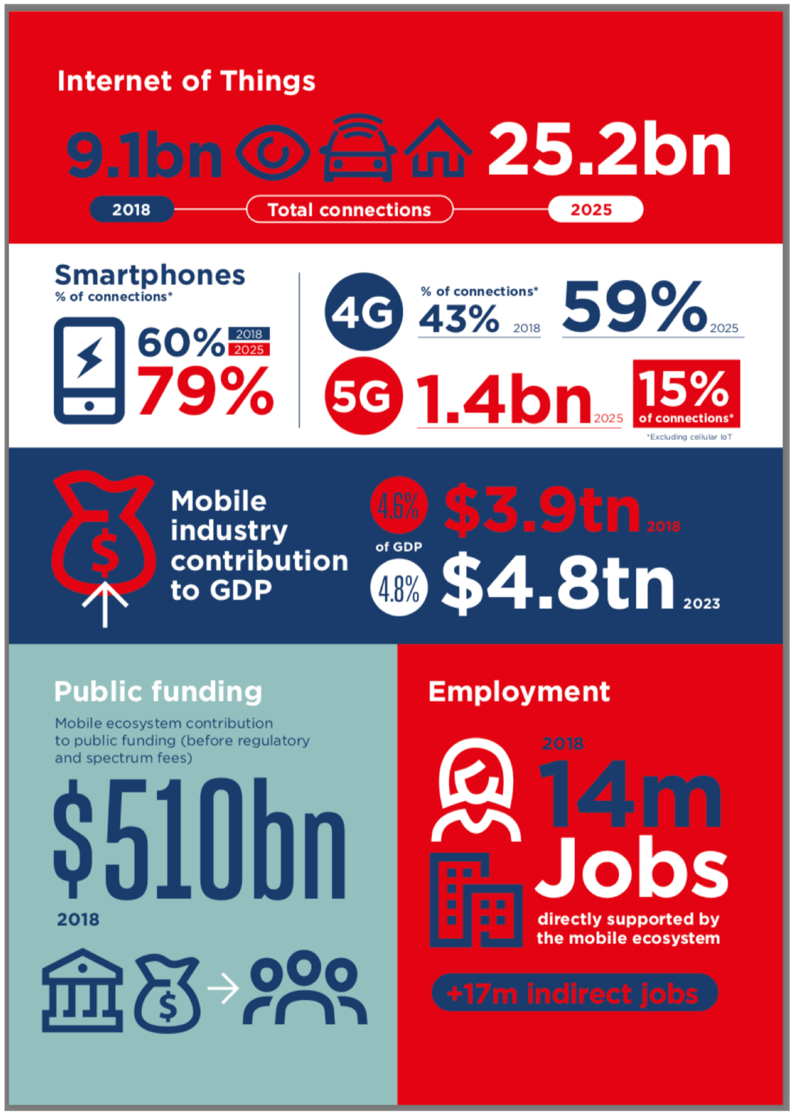
- GSMA represents the interests of mobile operators worldwide www.gsma.com

- GSMA Intelligence is the definitive source of global mobile operator data, analysis and forecasts www.gsmaintelligence.com

- Link to entire document:
  - https://www.gsma.com/r/mobileeconomy/

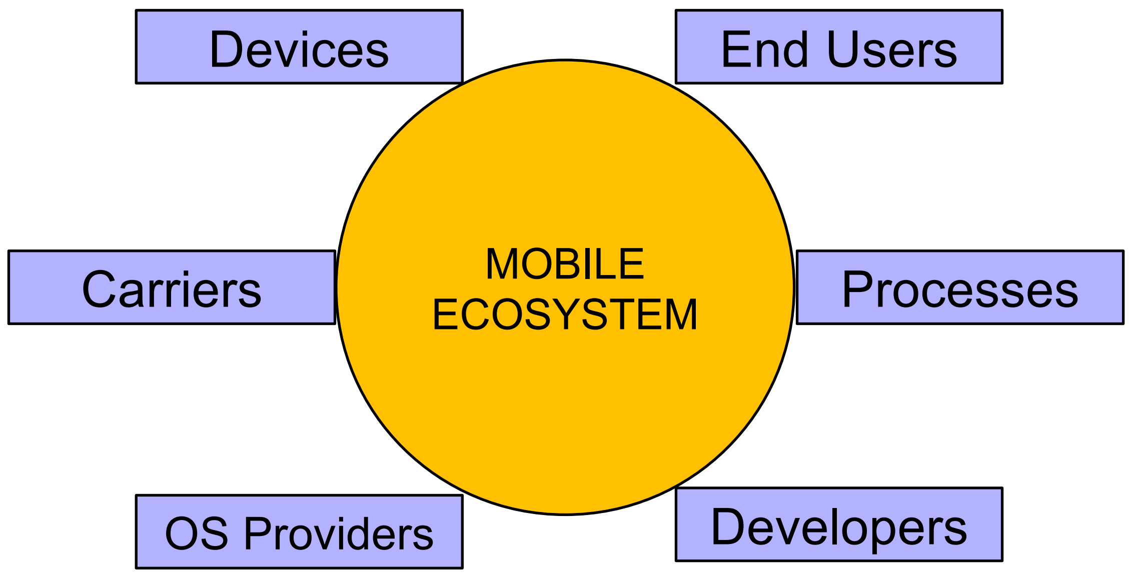
The Mobile Economy 2019





Mobile Ecosystem

Mobile Ecosystem is a collection of multiple devices, vendors, software, developers, processes, and users.



Mobile App Presentation

Features of mobile …

- It is *portable*
  - A mobile device can be carried anywhere.
- It is *personal*
  - Don't touch my phone!
  - Mobile means it is personal; not the property of my family or company; User chooses all setup options such as ring tone, etc.
- It is your personal companion
  - A mobile device is with you anytime and anywhere.
  - You don't forget your wallet or keys when you leave in the morning...
  - You also don't forget your mobile device.

Mobile App Presentation

Features of mobile …

- It is easy and fast to use
  - Does not need to boot like a computer.
  - Can be used sitting, walking, even lying down.
- It has some type of network connection
  - Connected to the internet when it is needed.

Mobile App Presentation

Moreover …

- There is only one web...desktop and mobile both use the same network protocols
- Users want a similar experience on the mobile as they have on their desktop
  - Many users always go to the desktop version even though they have to zoom and scroll...Why?
  - Maybe because the mobile version does not offer an acceptable user experience.

Mobile App Presentation

Moreover …

- Responsive Web Design using CSS Media Queries to create versions for all devices is becoming standard practice
  - Responsive Images (flexible or fluid images) use techniques such as CSS resizing, supplying different image sizes on the server side, or cropping images using CSS to adapt to different screen sizes and also different orientations.
- RESS: Responsive Web Design + Server Side Components
  - The most compatible version is determined on the server side and then RWD on the client side makes the final adjustments.

Mobile App Presentation

Moreover …

- Mobile First!
  - Concept where you design the mobile website first then design the desktop version. By designing mobile first, you are forced to think about the user interface, main areas needed, user input, etc. which will make both sites optimized and more usable.

Mobile App Presentation

Difference between desktop and mobile…

- Higher latency (slower)
- Screen size is smaller
- Less available memory
- Shorter battery life
- Testing and debugging is a challenge
- Browsers can behave differently
- Mobile users are many times in a distracted environment or 'on the move'
- Mobile users usually have immediate needs

Mobile App Presentation

Mobile Navigation Methods & ZOOM...

- *Focus* Navigation
  - Generally, a border or a background color is used to show the user where the focus is.
  - Normally used in nontouch devices where the user uses the cursor or keypad to navigate.
- *Cursor* Navigation
  - Emulates a mouse cursor by using the arrow keys. The Enter key emulates the mouse click.
- *Touch* Navigation
  - Most common method with most smart phones having touch screens.
  - Design is critical ... precision is much lower if fingers are used

Mobile App Presentation

Mobile Navigation Methods & ZOOM…

- *Basic* Zoom
  - Web page is rendered at a 1:1 scale to the original design
  - The scroll bar comes into play if the design does not fit on the screen
- *Smart* Zoom
  - Web page can be viewed at any zoom scale ... affects the font size, images, and the web page as a whole
  - Zoom is based on user gestures or menu options (ex. user can switch from full-page view to a paragraph view when the user double-taps on a specific paragraph)

Mobile App Presentation

Web Engines

- The **rendering engine** is the core of the browser which is the native code that parses and renders the page after reading the HTML, CSS and executing JavaScript code.
- WebKit is the main layout rendering engine used in Apple's Safari and Google Chrome browsers (Blink – a fork of Webkit in 2013) and other mobile devices (as of Mar 2018 71% of mobile browser market)...almost everyone uses WebKit so even different mobile devices can expect similar rendering.

Mobile App Presentation

Things (among others) that affect web browsing in the mobile environment...

- Display
  - The small screen does make a big difference...mobile web design cannot just be a smaller version of the desktop design
- Resolution
  - Resolution is a primary concern in mobile design. How many pixels are available? There are no mobile device standards regarding screen resolution. Touch-only devices normally have a higher resolution than those devices with a keyboard.

Mobile App Presentation

Questions to ask when defining and designing a mobile site…

- Why is the user accessing your mobile site? What is the user looking for?
- What are you offering from a mobile perspective that can solve the user's problem?
- What features do the users want?
- Where will the user be when accessing your site?
- Will a version be needed for older phones?
- Will I need to know where my user is located?

Mobile App Presentation

Best Practices…

- Keep It Simple (this does not mean Keep It Ugly!)
- Most frequent should be displayed first for mobile
- Desktop versions can handle alphabetical order because of larger screen size
- Should be no more than 3 levels of navigation
- Reduce user input to a minimum and use input defaults where possible
- Avoid horizontal scrolling and pop-up windows
- Provide navigation (also a Go To Top in footer if vertical scrolling is used)

Mobile App Presentation

Best Practices…

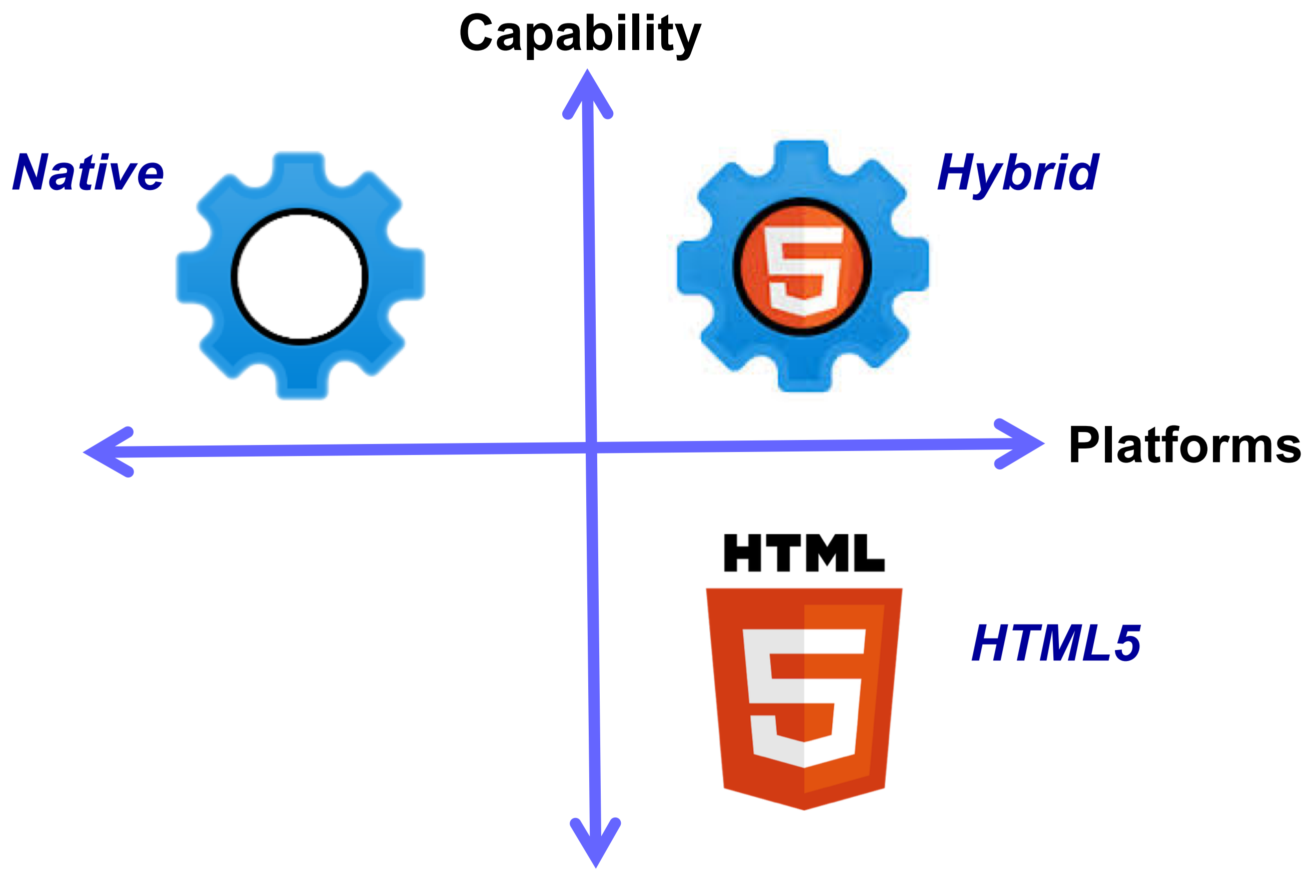
- Maintain consistency across pages...fonts, colours, etc.
- User Responsive Web Design for orientation changes
- Provide visual separation of sections
- Save user settings
- For colour, consider things such as sun, shade, contract and brightness
- Include accessibility support
- Testing should be done on actual devices if possible

Mobile App Development

Three types of Mobile Apps:

- Mobile Web (HTML5) Apps
  - Use standard web technologies—typically HTML5, JavaScript and CSS.
- Native Apps
  - Use the development tools and languages specific to a given mobile platform (e.g., Xcode and Objective-C with iOS, Eclipse and Java with Android).
- Hybrid Apps
  - Use embedded HTML5 apps inside a thin native container - combining the elements of native and HTML5 apps.

Mobile App Development



Mobile App Development

Web App Pros

- Allows wide coverage over multiple platforms
- Speed to market - quicker development
- Majority of code can be shared between all operating systems
- Well-known technologies used (HTML, CSS, Javascript, ...) - a wealth of technical resources
- Web APIs can be used - no waiting for a mobile SDK to appear
- Can be distributed freely or can be sold in vendor stores
- Can be self-updated

Mobile App Development

Web App Cons

- Porting is still required between platforms
- Debugging can be painful (but getting better)
- Performance will always be worse than native applications
- Not suitable for some applications/games
- Cannot create background applications on most platforms
- Cannot access all hardware and OS services

Examples of Mobile Websites

http://neatdesigns.net/27-superb-examples-of-mobile-optimized-websites/

JavaScript Review

Simple HTML5 page

```
<!DOCTYPE html>
<html>

  <head>
      <meta charset="utf-8">
      <title>HTML Example</title>
      <script src="myfile.js"></script>
  </head>
  <body>
     <header>  </header>
     <script> document.write('Hello world');</script>
     <section> </section>
     <footer> </footer>
  </body>

<html>
```

Recommended as it makes the code easier to read and maintain

There are two ways to embed JavaScript in HTML

Basic I/O

- `alert(…);`
- `console.log(…);`
- `confirm(…);`
- `prompt(…);`
- `document.write(…);`

Variables

- To Create a variable the var keyword is used followed by a case sensitive name **identifier.**
- Must begin with a letter, underscore (_), or dollar character ($).
- Cannot begin with a number or any other character that is not a letter (other than the underscore and dollar).
- Avoid keywords and reserved words.
- Can hold value of any type such as number, string, array, boolean, object, etc.
  - `var lastName, firstName = "Wilson";`
  - `var stdID; stdID = 2288900122;`

  -
    ```
    var firstTime = true, lastTime = false,
        empty = null;
    ```

Special Numeric Values

- Infinity, -Infinity, and NaN.
- Used to signal exception during a calculation.
  - `console.log(1 / 0);     // "Infinity"`
  - `console.log(-1 / 0);    // "-Infinity"`
  - `console.log(0 / 0);     // "NaN"`
  - `console.log("Hi" * 3); // "NaN"`

Operators

Common Arithmetic Operators

Symbol	Operator
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus (gets the remainder after division)
++	Increment (adds 1)
--	Decrement (subtracts 1)

Operators

Let’s practice…

`var x = 3, y = 4, z = x + y;`	`z = ??`
`var i = 7, j = 8, t = i % j;`	`t = ??`
`var nn = x * i;`	`nn = ??`
`var xx = x + i;`	`xx = ??`
`var s = "3", yy = s + 7;`	`yy = ??`
`document.write(yy*i);`	`result = ??`
`var zz = "good" + 7;`	`zz = ??`
`document.write(zz*x);`	`result = ??`

Operators

Common Comparison Operators

Symbol	Operator
==	Equal to
!=	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
===	Strictly equal to (both value and type must equal)
!==	Strictly not equal to (either value not equal or not of the same type)

Operators

Let's practice…

- `var x = 3, y = 4;` `(x > y) = ??`
- What is the output of this?

  -
    ```
    for (i = 0; i = 4; i++)
        alert(i);
    ```
- What does this code do?
  - `var y = 5;`

  -
    ```
    for (i = 0; i < y; i++)
        alert(i);
    ```
- `var i = "7", j = 7;` `(i == j) = ??`
- `var x = "80", y = 80;` `(x === y) = ??`

Math Methods

Common Math Method

Method	Description
abs(x)	Returns the absolute value of x
ceil(x)	Returns x, rounded upwards to the nearest integer
floor(x)	Returns x, rounded downwards to the nearest integer
max(x,y,.,n)	Returns the number with the highest value
min(x,y,.,n)	Returns the number with the lowest value
pow(x,y)	Returns the value of x to the power of y
random()	Returns a random number between 0 and 1
round(x)	Rounds x to the nearest integer

Type Conversion

- parseInt() returns the integer equivalent of its argument (or NaN)
- parseFloat() returns the float equivalent of its argument (or NaN)
- isNaN() return true if the argument cannot be converted to a number.

```
parseInt("39");               parseInt("batman888")
parseInt("+007Bond");         parseInt("-57ooo");
parseFloat("28");             parseFloat("62.32");
parseFloat("9.9.9");          parseFloat("-17.88");
```

Reading assignments

Please read about the following topics

- loop (while , for , for in, for of)
- if...else Statement
- Switch Case
- Loop Control (continue and break**)**

Arrays

Arrays in JavaScript are typeless which means you can mix data types within an array.

- 1- Numeric array (An array with a numeric index)

```
var cities = new Array("Toronto","London","Brampton");
var cities = ["Toronto", "London", "Mississauga"];
console.log(cities[0]);  // Toronto
```

- 2- Associative Array (An array with strings as index)
  - uses a key string to identify an element instead of a numeric index.

```
var user = [];
user["name"] = "Tim";
user["age"] = 25;
document.write(user.name + " is " + user.age);
// "Tim is 25"
```

Array Object Properties / Methods

```
var cities = new Array("Toronto", "London", "Brampton" );

var len = cities.length; // 3
```

Method	Example	Description
concat()	array3 = array1.concat(array2)	Joins two or more arrays
indexOf()	array.indexOf("xxx")	Search the array for an element and returns its position
join()	array.join()	Joins all elements of an array into a string
lastIndexOf()	array.lastIndexOf("xxx")	Search the array for an element, starting at the end, and returns its position

Array Object Properties / Methods

Method	Example	Description
pop()	array.pop()	Removes the last element of an array
push()	array.push("xxx")	Adds new element to the end of an array
reverse()	array.reverse()	Reverses the order of the elements in an array
shift()	array.shift()	Removes the first element of an array
sort()	array.sort()	Sorts the elements of an array

Object Arrays

```
function Students(id, firstName, lastName)  {
   this.id = id;
   this.firstName = firstName;
   this.lastName = lastName;   }

var students = new Array( );
students.push(new Students("123", "John", "Dow"));
students.push(new Students("234", "Anshul", "Kaur"));
students.push(new Students("999", "Phil", "Green"));

for (var i=0; i<students.length; i++)
   console.log(students[i].id + "  " +
               students[i].firstName + "  " +
               students[i].lastName);
```

Practice

1. Write a script to define an array.
2. Add the following data to the array:

•	Andy,	Pak,	Brampton,	9058892929
•	Wilson,	Smith,	Toronto,	4160023915
•	Praba,	Singh,	Mississauga,	9056627123
•	Jaspreet,	Gill,	Toronto,	4167726777
•	Arthur,	Sung,	Oakville,	9059620123

3. List the data from the array to the console.

Functions

Reusable code blocks that will only execute when called

```
function sayHello() {   // function definition
       alert("Hello world");
}

sayHello(); // invoke the function
```

Function as expression:

```
var greet = function sayHello() {   // function definition
       alert("Hello world");
};

greet();    // invoke the function
```

Functions

Anonymous Function:

```
var greet = function() {    // function definition
       alert("Hello world");
};
```

Return Statement causes the function to exit and return the specified value.

```
function foo() {
      return; // exit function
}

console.log( foo() ); // "undefined"
```

Variable Scope and Lifetime

- JavaScript doesn’t require variables to be declared.
- Variables not declared will have global scope.
- All variables declared inside functions or listed as parameters are local to the function.

```
var xx = 0;
function sum(zz) {
    var xx = 1;
    yy = xx + zz;
    console.log(xx + " " + yy + " " + zz);
}
sum(50);
```

What are xx, yy, zz before and after running the function?