LIFERAY®

MODERN WEB EXPERIENCES: ECMASCRIPT 2015 FEATURES

ECMASCRIPT 2015

- *ECMAScript 2015*, previously known as ES6, is the latest version of the ECMAScript standard.
- In DXP, you can now write JavaScript that adheres to the new ECMAScript 2015 syntax, leverage ES2015 advanced features in your modules, and publish them.
- ECMAScript 2015 is a significant update to the language, and the first update to the language since ES5 was standardized in 2009.

```
export function sum(x, y) {
  return x + y;
}
export var pi = 3.141593;
```

WHAT'S NEW IN ECMASCRIPT 2015?

- Here is a short list of some of the benefits of using ECMAScript 2015:
  - Class syntax like other OO languages
    - Classes support prototype-based inheritance, super calls, instance and static methods, and constructors.
  - Arrow method syntax
    - `var odds = numbers.map(v => v + 1);`
  - Modules
  - `Let` and `const` declarations
  - Language-level support for modules for component definition
    - Codifies patterns from popular JavaScript module loaders and specifications like asynchronous module defintion (AMD).
- Let's take a look at examples of each of these features.

CLASSES

- **Classes** with constructors and inheritance:

```
class Car {
    constructor(make) { //constructors!
        this.currentSpeed = 25;
    }

    printCurrentSpeed(){
         console.log('current speed: ' + this.currentSpeed + ' mph.');
    }
}
class RaceCar extends Car { //inheritance
    constructor(make, topSpeed) {
        super(make);
        this.topSpeed = topSpeed;
    }

    goFast(){
         this.currentSpeed = this.topSpeed;
    }
}
```

ARROW FUNCTIONS

- **Arrow Functions**, which make anonymous functions easier:

```
setTimeout(() => {
    alert("Hello from an arrow function!")
}, 1000);
```

MODULES

- **Modules** give you the ability to create, load, and manage dependencies via the new import and export keywords:

```
import $ from 'lib/jquery';
```

- To make modules discoverable, you need to write a `package.json` file with the name and version of your ECMAScript 2015 module.

LET AND CONST DECLARATIONS

- **let** and **const** declarations:

```
//let
function letTest() {
  let x = 1; // let declares a frame scope local variable
  if (true) {
    let x = 2;  // different variable
    console.log(x);  // 2
  }
  console.log(x);  // 1
}

//const
const ALWAYS_SEVEN = 7;

// this will throw an error
ALWAYS_SEVEN = 8;
```

ECMASCRIPT 2015 BROWSER SUPPORT

- Most modern browsers support ECMAScript 2015. Liferay DXP comes with the ability to transpile ECMAScript 2015 code.
- To use ECMAScript 2015 syntax and advanced features, you need to make the following adjustments to your JavaScript files:
    1. Files containing ECMAScript 2015 code that needs to be transpiled should end in `.es.js`.
    2. Import the `polyfillBabel` class from the `polyfill-babel` module to use advanced features like generators.

       ```
       import polyfillBabel from 'polyfill-babel'
       ```
- With Themes, you can also take advantage of *Liferay Theme ES2015 Hook*: *https://www.npmjs.com/package/liferay-theme-es2015-hook*
- *ECMA-262 6th Edition, The ECMAScript 2015 Language Specification* can be found here: *http://www.ecma-international.org/ecma-262/6.0/*

Notes: