What is ES6

○ A javascript framework

●A javascript version

○ A new programming language

Below items are about javascript, which ones are new and specific to ES6?

● the keyword "class"

● Promises

● The keyword "const"

 ○ the keyword "new"

● string interpolation

● arrow functions

○ callbacks

○ regex expressions

● constants

Can I write the following in ES6 ?

const a= 1 ;

a= 2 ;

○ yes

● no

Can I write the following in ES6 ?

const a= {value: 1} ;

a.value= 2 ;

● yes

○ no

Destructuring Assignent is interesting for?

● Assign several variables at a time

● Assign variables based on javascript structures

What happens with the following code?

let { firstName, lastName } = person;

○ I’m assigning the firstName and lastName values **to the person fields**

● I’m using the person fields to assign values **to the firstName and lastName variables**

The following code allows to define default values for function parameters:

let createPerson= function(firstName, lastName){

let fName= firstName || 'ty',

lName= lastName || 'segall';

return new Person(fName, lName);

}

Is there an easier way to do that with ES6

● Yes

○ No

Consider the following code:

createPerson= function(firstName = 'Michelle', lastName){

console.log(firstName + ' ' + lastName);

}

createPerson(' Barack', 'Obama ');

What will be logged ?

○ Michelle Obama

● Barack Obama

Consider the following code

checkRestParams= function(firstArg){

let argsButFirst= arguments && (arguments.length>1) && [] || undefined;

if (argsButFirst){

for (let i= 1; i< arguments.length; i++){

argsButFirst.push(arguments[i]);

}

}

console.log(argsButFirst);

}

Is there a better way to do that with ES6?

● Yes

○ No

Consider the following code:

let restSample= function(first, ...others){

return others;

}

let [first, ...others] = ['a', 'b', 'c', 'd'];

console.log(others);

console.log(restSample(others));

What will be logged ?

○ ['a', 'b', 'c', 'd'] ['a', 'b', 'c', 'd']

○ ['b', 'c', 'd'] ['b', 'c', 'd']

○ ['b', 'c', 'd'] ['c', 'd']

● ['b', 'c', 'd'] undefined

Where will you initialize ES6 class fields?

○ In the class scope itself

● In the class constructor

Can an ES6 class implement an interface?

○ yes

● no

Can an ES6 class extend a super class?

○ no

● Yes

 What are the promises used for

● Deal with asynchronous operations

●  Simplify the code complexity when using lot of callbacks

● Chain asynchronous calls

○ Create factories

○ Help with generators

Why use promises instead of passing callback into functions?

○ faster at runtime

● code is easier to read and maintain

Consider the following code:

 new Promise( (resolve, reject) => resolve([1]))

.then( (data) => console.log(data) );

What will be logged?

○ 1

● [1]

What will be logged?

new Promise( (resolve, reject) => resolve([1]))

.then( (data) => data )

.then( (data) => console.log(data) );

○ Nothing

● [1]

○ 1

What will be logged?

new Promise( (resolve, reject) => resolve([1]))

.then( (data) => new Promise( (resolve, reject) => resolve(data) ) )

.then( (data) => console.log(data) );

○ Nothing

● [1]

○ 1

What will be logged?

new Promise( (resolve, reject) => resolve([1]))

.then( (data) => new Promise( (resolve, reject) => resolve(data) ) )

.then( (data) => console.log() )

.then( (data) => console.log(data) );

● Nothing

○ [1]

○ 1

Why are arrow functions interesting?

○ They allow to create anonymous functions

○ They allow to create callbacks

● They provide a shortcut syntax to return a single expression

● They have a more intuitive understanding of the “this” keyword

Check the correct affirmations about ES6 modules

● They encourage to split the code into files rather than getting a big file

○ They allow to load less code at runtime

● They allow to organize the code differently, which makes it easier to read / maintain

○ You can only export one value from a given module

● You can import as many values as you want from external modules

○ If modules A and B both export the value ‘a’, there is no way to import ‘a’ twice from a third module because they have the same name (need an intermediary module to rename one value before importing it)