JavaScript QCM 9 Adrien JOLY

Bonjour, Adrien JOLY

Note obtenue: 7 / 7

Sign out

QCM

Question 1 (1 points)

Qu'est-ce qu'un composant ?

- ○ c'est le nom qu'on donne à tout programme JavaScript
- ○ c'est le nom qu'on donne à tout div d'une page HTML
- ◉ c'est un module qui a été conçu pour être intégré par d'autres développeurs
- ○ c'est une page web de documentation

Un composant est un module qui a été conçu pour être intégré par d'autres développeurs.

Sur le Web, un composant est généralement composé d'un script JavaScript, et éventuellement d'un fichier CSS en plus. Il est aussi accompagné d'une page qui explique aux développeurs comment intégrer ce composant sur leur page.

Question 2 (1 points)

Nous avons vu en cours que tout composant devait être conçu de manière à pouvoir être intégré plusieurs fois sur une même page.

Parmi ces stratégies de conception, laquelle ne permet PAS l'intégration multiple d'un composant:

- ○ Faire en sorte que le composant s'applique des éléments groupés
- ○ Demander à l'intégrateur d'appeler une fonction pour chaque intégration
- ◉ Demander à l'intégrateur de modifier le code source du composant

Tout composant doit être conçu de manière à ce que l'intégrateur n'ait pas besoin de modifier le code source du composant, ni même de lire ce code source.

Afin que le composant puisse s'instancier plus d'une fois sur une même page, le développeur du composant peut s'y prendre de différentes manières, notamment:

- faire en sorte que le composant s'intègre à partir de groupes d'éléments. (exemple d'instructions: *chaque* `<div>` *contenant des* `<img>` *sera transformé en galerie*, au lieu de *toutes les* `<img>` *de la page seront regroupées dans une même galerie*)
- ou demander à l'intégrateur d'appeler une fonction pour chaque intégration. (exemple d'instructions: *pour chaque galerie à intégrer sur votre page, appeler la fonction* `creerGalerie()` *en passant le* `<div>` *concerné en paramètre*)

Question 3 (1 points)

Soit le fichier HTML suivant:

```
<ul>
  <li data-number="1">premier</li>
  <li data-number="2">deuxième</li>
</ul>
```

Comment récupérer la valeur de l'attribut `data-number` du deuxième élément `<li>` ?

- ○ document.getElementById('2').data-number
- ○ document.getElementsByTagName('li')[1].data-number
- ○ document.getElementsByTagName('li').getAttribute('data-number')[1]
- ◉ document.getElementsByTagName('li')[1].getAttribute('data-number')

- pour accéder au deuxième élément `<li>`, il faut utiliser la fonction `getElementsByClassName()` puis accéder au 2ème élément du tableau retourné par cet appel à l'aide de `[1]`
- seuls les attributs standard définis par le DOM (ex: id, href, src...) sont accessibles directement par des propriétés d'objet de classe Element, via la notation pointée.
- comme `data-number` n'est pas une propriété standard, il faut appeler la méthode `getAttribute()` sur l'objet de classe Element pour y accéder.

Question 4 (1 points)

```
var lien = document.getElementById('mon-lien-a');
```

Comment changer le titre du lien `<a>` (tel qu'il sera affiché à l'écran) référencé par la variable `lien` ?

- ○ lien.title = 'nouveau titre';
- ◉ lien.innerHTML = 'nouveau titre';
- ○ lien.href = 'nouveau titre';
- ○ lien.setAttribute('title', 'nouveau titre');

- en HTML, le titre visible d'un lien est défini par le contenu de l'élément (ce qu'il y a entre les balises `<a>` et `</a>`); il ne faut donc pas modifier un attribut de ce lien (ex: `title` et `href`) mais son contenu.
- pour modifier le contenu d'un élément, il faut utiliser la propriété `innerHTML`.

Question 5 (1 points)

Soit le code JS suivant:

```
// supposons que boutons référence un ensemble de boutons
for (var i = 0; i < boutons.length; i++) {
  boutons[i].onclick = function() {
    alert('vous avez cliqué sur le bouton nº' + i);
  }
}
```

Que se passera-t-il quand on cliquera sur chaque bouton ?

- ○ on verra l'alerte: "vous avez cliqué sur le bouton n°i"
- ○ le numéro du bouton cliqué apparaitra dans un alert
- ◉ un numéro de bouton incorrect apparaitra dans un alert
- ○ il ne se passera rien

réponse: un numéro de bouton incorrect apparaitra dans un alert

Question 6 (1 points)

Expliquez la raison du comportement observé à la question ci-dessus.

- ○ il y a une erreur de syntaxe dans le code
- ◉ au moment du clic, la variable i vaudra boutons.length
- ○ au moment du clic, la variable i vaudra 0
- ○ la variable i n'est pas accessible dans le contexte de la fonction onclick

- Au moment du clic sur un élément, le navigateur appelle la fonction rattachée à sa propriété `onclick`. Notre fonction `onclick` a bien accès à la variable `i`, car cette fonction est définie dans le même contexte (*scope*) que celui dans lequel cette variable a été créée.

- Par contre, lorsque le navigateur appellera cette fonction (au moment où l'utilisateur aura cliqué sur un bouton), la boucle aura déjà fini d'itérer, et valeur de `i` aura donc atteint sa valeur maximale: `boutons.length`.

Question 7 (1 points)

Quelle est la meilleure manière de résoudre le problème de la question ci-dessus ?

- ○ il faut ralentir les itérations de la boucle
- ○ il faut écrire une fonction onclick par bouton
- ○ il faut passer la valeur de i en paramètre de notre fonction onclick
- ◉ il faut passer la valeur de i en paramètre d'une autre fonction, à chaque itération de la boucle

Pour que la fonction `onclick` qu'on définit pour chaque bouton conserve la valeur de la variable `i`, il faut que cette fonction soit définie dans un contexte où la valeur sera stockée dans une variable séparée.

Vu que, en JavaScript, chaque variable est rattachée à la fonction dans laquelle elle est créée, la bonne pratique consiste à définir une fonction qui prendra la valeur de `i` en paramètre, puis retournera la fonction `onclick` qui utilisera cette valeur.