Commencé le	mercredi 15 février 2023, 18:02
État	Terminé
Terminé le	mercredi 15 février 2023, 19:05
Temps mis	1 heure 2 min
Note	**22,80** sur 25,00 (**91,2**%)

Description

LISEZ CES INSTRUCTIONS AVANT DE COMMENCER L'EXAMEN

- L'examen est noté sur un total de 25 points.
- La note partielle associée à chaque question est marquée à gauche de la question.
- Certaines questions demandent d'ajouter une justification à votre réponse. Une bonne réponse sans justification valide ne sera pas acceptée.
- **Vous avez une seule tentative pour l'examen! Ne soumettez pas votre tentative à moins d'être 100% sûr(e) d'avoir terminé l'examen ! La soumission sera automatique à la fin de la période.**
- **En cas de doute sur le sens d'une question, faites une supposition raisonnable, énoncez-là clairement dans votre réponse et poursuivez. Une "question" vide est disponible sur la première page d'examen si vous avez besoin de plus de place ou pour des questions spécifiques.**
- **Vous avez accès à une copie des notes de cours PDF sur l'instance Moodle et droit à 1 feuille recto-verso (imprimée ou manuscrite) comme documentation.**

Bon travail!

Question **1**

Terminé

Non noté

Cette question est un espace dédié à vos commentaires ou questions sur l'examen. Nous ne répondons à aucune question pendant l'examen.

Priorisez de mettre vos commentaires et/ou hypothèses directement dans la question spécifique.

Pour la question 9, Je considère qu'on peut installer une implémentation client de WebSocket sur le serveur qui pourra alors aller demander au service externe de trouver les différences entre les images lorsqu'il reçoit ces deux images du client. Le serveur se connectera donc au service externe et une fois les différences reçues, il renverra l'information requise au client. Il faudra bien sûr s'assurer de gérer les erreurs qu'il peut y avoir lors de la requête (ex. service externe hors-ligne) et renvoyer un message d'erreur au client dans ce cas.

By the way, c'est quoi le STEP ?

Question **2**

Terminé

Note de 3,50 sur 5,00

Voici un extrait du code du gabarit HTML de votre classe **GameCarouselComponent** qui possède comme enfants des instances de **GameCardComponent**.
Le tableau **gameCards** possède les jeux à afficher (1 à 4) sous la forme d'un objet **GameCard** qui contient tous les attributs nécessaires pour l'affichage.
GameCardComponent possède un seul attribut public qui est : @Input() gameCard: **GameCard**

```
1 <div id="game-container" *ngIf="gameCards.length > 0">
2     <div id="game-container-2" *ngFor="let gameCard of gameCards">
3         <app-game-card gameCard="gameCard"> </app-game-card>
4     </div>
5 </div>
```

a) Votre collègue vous demande pourquoi est-ce qu'il y a 2 éléments qui englobent <app-game-card>. Que lui répondez-vous ? Est-ce qu'il y a une manière plus simple qui permet d'avoir le même comportement ? Justifiez votre réponse. (2 points)

b) Le compilateur d'Angular lance une erreur lors de la transpilation de la ligne 3 du gabarit. Pourquoi ? (1 point)

c) **GameCarouselComponent** récupère les informations de tous les jeux disponibles du serveur d'un seul coup au chargement de la page, mais charge seulement 4 jeux à la fois dans le tableau **gameCards**. Donnez un avantage et un désavantage de cette approche dans votre système. (2 points)

a) Le premier div permet d'afficher les jeux uniquement si leur nombre est supérieur à 0 (uniquement s'il y a des jeux). Le deuxième div permet d'itérer à travers chaque jeu pour l'afficher avec son propre component. Une manière plus simple d'avoir le même comportement serait simplement d'enlever le premier div car si un fait une boucle sur un array vide, elle va s'exécuter 0 fois de toute façon et ne rien afficher.

b) Pour pouvoir utiliser une variable d'un component comme prop dans le html, il faut plutôt écrire `[gameCard]="gameCard"`. Sinon, c'est le string "gameCard" qui est envoyé au component **GameCardComponent** alors que celui-ci s'attend à recevoir un prop de type **GameCard**.

c) Un avantage de charger tous les jeux d'un coup au chargement de la page est qu'il n'y aura pas de chargements et donc de délais supplémentaires lorsqu'on passera à la page suivante, les jeux seront déjà en mémoire, et on pourra afficher les nouvelles informations instantanément. Par contre, cela signifie que si le nombre total de jeux est élevé, le temps requis avant l'affichage de la première page pourrait être élevé car il faut charger l'ensemble des jeux depuis le serveur dès le début.

Commentaire :
a) Les sont nécessaires puisqu'on ne peut pas avoir *ngIf et *ngFor sur le même élément -1

<app-game-card *ngFor="let gameCard of gameCards"> : manière plus simple avec *ngFor directement sur le component -0.5

b) ok

c) ok

Question **3**

Correct

Note de 1,00 sur 1,00

Vous avez un Service qui possède un attribut de type Subject<string> qui est utilisé pour communiquer avec plusieurs de vos Components qui s'y abonnent à travers la méthode *subscribe()*.

Pourquoi est-il important de se désabonner avec la méthode *unsubscribe()* de l'Observable à un moment donné du cycle de vie de chaque Component ?

a. La librairie RxJS force l'utilisation des méthodes *subscribe()* et *unsubscribe()* pour les Observables.

b. Un Observable peut être observé par un seul Observer à la fois. Le désabonnement permet aux autres Observers de fonctionner.

c. La méthode *unsubscribe()* n'est pas nécessaire puisque RxJS est capable de gérer le désabonnement automatiquement pour vous.

d. Un abonnement non terminé pourrait causer une fuite de mémoire. ✔

Votre réponse est correcte.

La réponse correcte est :
Un abonnement non terminé pourrait causer une fuite de mémoire.

Question **4**

Correct

Note de 1,00 sur 1,00

Voici l'implémentation de votre **ChronometerService** qui gère la minuterie dans vos parties. Ses 2 méthodes sont appelées au début et à la fin de chaque partie et l'attribut *nbElapsedSeconds* est affiché à l'écran dans votre **GamePageComponent**. La fonction *interval* est un Observable qui émet à chaque seconde (1000ms).

Quelle sera la valeur initiale du temps affichée lorsque vous débutez une 2e partie après avoir complété une partie ayant pris 30 secondes ?

```
@Injectable({ providedIn: 'root'})
export class ChronometerService {
   public nbElapsedSeconds: number = 0;
   private chronometer: Subscription = new Subscription();
   public startChronometer(): void {
       this.chronometer = interval(N_MS_IN_SECOND).subscribe(() => {
           this.nbElapsedSeconds++;
       });
   }
   public stopChronometer(): number {
       if (this.chronometer) this.chronometer.unsubscribe();
       return this.nbElapsedSeconds;
   }
}
```

- a. 30 secondes ✔
- b. Aucune de ses réponses
- c. undefined
- d. 30 secondes + le temps écoulé entre la fin de la 1re partie et le débu de la 2e partie
- e. 0 secondes

Votre réponse est correcte.

La réponse correcte est :
30 secondes

Question **5**

Partiellement correct

Note de 0,55 sur 1,00

Parmi les énoncés suivants, lesquels sont erronées ?

- [x] a. L'utilisation de Services est obligatoire dans un projet Angular. ✔
- [x] b. L'utilisation d'un Service élimine le besoin d'avoir un lien parent/enfant entre les Components. ✔
- [x] c. Les décorateurs @Input/@Output permettent le passage de valeurs primitives seulement, contrairement aux Services qui permettent la communication d'objets complexes. ✔
- [ ] d. La syntaxe [(maVariable)] est la seule manière de faire une liaison bi-directionnelle entre un Gabarit et le code TS de son Component.
- [ ] e. Un Component est identifié seulement par son attribut 'selector' dans le gabarit HTML d'un autre Component.
- [x] f. La directive *ngIf ne peut pas être placée sur le même élément HTML que la directive *ngFor. ✖

Votre réponse est partiellement correcte.

Vous en avez sélectionné correctement 3.
Les réponses correctes sont :
L'utilisation de Services est obligatoire dans un projet Angular.,

La syntaxe [(maVariable)] est la seule manière de faire une liaison bi-directionnelle entre un Gabarit et le code TS de son Component.,

Les décorateurs @Input/@Output permettent le passage de valeurs primitives seulement, contrairement aux Services qui permettent la communication d'objets complexes.,

L'utilisation d'un Service élimine le besoin d'avoir un lien parent/enfant entre les Components.

Question **6**

Correct

Note de 1,00 sur 1,00

Quelle est la différence entre Express, NodeJS et Socket.IO ?

- ( ) a. La librairie Express est nécessaire pour traiter des requêtes HTTP, mais Socket.IO est optionnel pour un serveur utilisant NodeJS.
- (•) b. NodeJS est un environnement d'exécution à partir duquel on utilise les librairies Express et Socket.IO, pour une gestion à plus haut niveau des requêtes HTTP et de la communication WebSocket, respectivement. ✔
- ( ) c. NodeJS est un environnement d'exécution à partir duquel on utilise la librairie Express pour une gestion à plus haut niveau des requêtes HTTP et SocketIO est un protocole de communication bidirectionnelle.
- ( ) d. NodeJS, Express et Socket.IO sont des librairies côté serveur pour la communication réseau.
- ( ) e. La librairie Socket.IO est nécessaire pour traiter le protocole WebSocket, mais Express est optionnelle pour un serveur utilisant NodeJS.

Votre réponse est correcte.

La réponse correcte est :
NodeJS est un environnement d'exécution à partir duquel on utilise les librairies Express et Socket.IO, pour une gestion à plus haut niveau des requêtes HTTP et de la communication WebSocket, respectivement.

Question **7**

Terminé

Note de 3,75 sur 4,00

Voici l'implémentation de la gestion du clavardage dans plusieurs salles de votre projet.
Vous avez présentement 3 clients qui communiquent avec votre serveur : ClientA, ClientB et ClientC.
ClientB et ClientC sont présentement dans la même partie de jeu et dans la même *Room* ayant l'identifiant "XYZ".

Voici la gestion de cet événement du côté serveur. La fonction *getRoomFromSocketId(socketId)* retourne l'identifiant de la salle d'un socket dans une partie de jeu :

```
socket.on('chat', (message) => {
  const room = this.getRoomFromSocketId(socket.id);
  const chatMessage = { room: room , ...message}
  socket.broadcast.emit("chatMessage", chatMessage);
});
```

Voici également la gestion du message du serveur du côté client. L'attribut *roomId* représente le nom de la Room dans laquelle le client est présentement :

```
this.socketService.on('chatMessage', (chatMessage) => {
  if (chatMessage.room === this.roomId) {
    this.chatMessages.push(chatMessage);
  }
});
```

ClientB vient d'envoyer l'événement "chat" avec l'objet **{ message: "Salut", player: "ClientB" }** au serveur.

a) En fonction de la configuration donnée, qui recevera un message du serveur et **pourquoi** ? (1.5 points)

b) Cette implémentation possède un problème avec la gestion des messages dans les salles. Donnez le problème avec l'implémentation et proposez une solution possible. (2.5 points)

a) Les clients qui recevront un message du serveur sont **ClientA** et **ClientC** à cause du `socket.broadcast.emit()` qui envoie le message à tout le monde sauf l'émetteur. Par contre ce sera seulement **ClientC** qui va traiter le message à cause de la condition if dans le code du client.

b) Le problème de cette implémentation est que les messages du chat sont envoyés à tous les clients, et c'est ensuite au code du client de décider si le message est affiché ou pas. Si un utilisateur décide d'inspecter les échanges entre son navigateur et le serveur, il verra tous les messages envoyés par les autres clients, peu importe la room. Une solution serait de seulement envoyer les messages aux clients connectés à la même room que l'émetteur. Par exemple en remplaçant la ligne 4 du code du serveur par `io.to(room).emit(chatMessage)`. On pourrait ensuite enlever la condition `if` dans le code du client puisque ce serait le serveur qui ferait en sorte que les clients ne reçoivent que les messages qui les concernent. Il faudrait seulement ajouter un check côté client pour ne pas afficher les messages qui ont été envoyés par le client lui-même pour éviter que les messages ne soient envoyés en double.

Commentaire :
b) -0.25 la réponse ne mentionne pas le fait que l'objet chatMessage est maintenant non nécessaire vu que l'attribut "room" ne devrait pas être envoyé au client

Question **8**

Correct

Note de 1,00 sur 1,00

La création d'une nouvelle fiche de jeu dans votre système se fait à travers une requête POST sur la route /games/ et les informations de la fiche dans le corps de la requête.

Vous avez décidé de ne pas permettre la création de 2 fiches avec le même nom. Lorsqu'un tel cas arrive, le code de retour dans la réponse HTTP est : 404 Not Found.

Sémantiquement, quel est le **meilleur** code de réponse dans une telle situation ?

- a. 500 Internal Server Error
- b. 204 No Content
- c. 409 Conflict ✔
- d. 404 Not Found
- e. Aucune de ses réponses
- f. 400 Bad Request

Votre réponse est correcte.

La réponse correcte est :
409 Conflict

Question **9**

Correct

Note de 1,00 sur 1,00

Vous êtes responsables de l'implémentation de la logique du système de détection de différences ente 2 images pour le Sprint 1 et vous avez trouvé un service en ligne qui permet de générer les différences entre 2 images avec un rayon d'élargissement. Vous voulez l'utiliser pour générer les images de différences pour vos jeux.

Votre équipe a déjà mis en place une communication utilisant Socket.IO pour la logique de jeu entre votre site web et votre serveur. Cependant, le serveur du service trouvé n'utilise pas Socket.IO, mais plutôt l'implémentation native de WebSocket pour sa communication.

Pouvez-vous quand même utiliser ce service dans votre projet dans son état actuel ?

- a. Oui, mais vous ne pouvez pas utiliser les fonctionnalités supplémentaires que Socket.IO offre par-dessus WebSocket.
- b. Non, votre serveur ne peut pas communiquer avec un autre serveur, seulement avec des clients.
- c. Oui, mais vous devez établir une connexion supplémentaire en utilisant le protocole WebSocket seulement. ✔
- d. Non, le client et le serveur doivent utiliser Socket.IO pour une communication valide.
- e. Oui, Socket.IO utilise le protocole WebSocket pour communiquer, donc les 2 systèmes sont compatibles.
- f. Oui, mais vous devez contacter le service à travers votre client puisque WebSocket est un protocole client-serveur.

Votre réponse est correcte.

La réponse correcte est :
Oui, mais vous devez établir une connexion supplémentaire en utilisant le protocole WebSocket seulement.

Question **10**

Correct

Note de 1,00 sur 1,00

Les tests du côté client utilisant les librairies Jasmine et Karma sont toujours exécutés dans un ordre aléatoire d'un lancement des tests à un autre. Pourquoi ?

a. Ceci n'a aucun impact sur les tests exécutés et leurs résultats.

b. Ceci est une limitation du fonctionnement de l'outil Karma.

c. Ceci permet de détecter la présence de dépendances entre les tests unitaires. ✔

d. Ceci permet d'exécuter les tests plus rapidement.

Votre réponse est correcte.

La réponse correcte est :
Ceci permet de détecter la présence de dépendances entre les tests unitaires.

Question **11**

Terminé

Note de 3,00 sur 3,00

Voici le constructeur du component **GameImageComponent** qui utilise la classe **GameService** ainsi que la configuration de ses tests unitaires.

```
constructor(private readonly gameService: GameService) {}

// game-image.component.spec.ts
beforeEach(async () => {
   gameSpy = jasmine.createSpyObj('GameService', ['handleClick','blinkDifference','playSound']);
   TestBed.configureTestingModule({
       declarations: [GameImageComponent],
       providers: [{ provide: GameService, useValue: gameSpy }],
   }).compileComponents();
});
```

a) Quel est le rôle de l'objet **gameSpy** dans la configuration des tests? Est-ce que cet objet est nécessaire ? **Justifiez** votre choix. (2 points)

b) Suite à l'exécution des tests, vous obtenez à plusieurs reprises le message d'erreur suivant dans la console : ERROR: 'NG0304: 'app-differences-area is not a known element:' .
Quelle est **la raison** de ce message ? (1 point)

a) L'objet gameSpy est nécessaire car il permet d'isoler le comportement du GameImageComponent, c'est à dire de tester uniquement son code et pas le code des services qu'il utilise. Lorsqu'on voudra tester une des fonctions de GameImageComponent, on pourra donner une fausse valeur de retour aux fonctions mockées de GameService pour voir quel est le comportement de notre component lorsqu'il reçoit cette valeur en retour à l'appel à la fonction de GameService. On pourra aussi vérifier que la méthode du service a vraiment été appelée avec `expect(gameSpy.handleClick).toHaveBeenCalled()` par exemple.

b) Ce message signifie que le gabarit HTML de GameImageComponent a dans ses éléments le component AppDifferencesArea. Lors des tests, on ne peut pas directement utiliser d'autre components réels, il faut les mock eux aussi.

Commentaire :
a) ok

b) ok

Question **12**

Terminé

Note de 4,00 sur 4,00

Considérez le code suivant qui gère le téléversement et l'affichage d'une image dans la Vue de création.

```
1  async imageUpload(e: Event, index: number): Promise<void> {
2    let f = (e.target as HTMLInputElement).files.item(0) as File;
3
4    const check = !(this.imagesService.isNotValidImage(f));
5    if (!check) {
6      this.displayImageError();
7      return;
8    }
9
10   switch (index) {
11     case 0: {
12     this.drawImage(this.originalCanvas.nativeElement, f);
13     break;
14     }
15     case 1: {
16     this.drawImage(this.modCnvs.nativeElement, f);
17     break;
18     }
19     case 2: {
20     this.drawImage(this.originalCanvas.nativeElement, f);
21     this.drawImage(this.modCnvs.nativeElement, f);
22     break;
23     }
24   }
25 }
```

a) Identifiez les erreurs de qualité de code présentes. Considérez que les autres fonctions appelées sont bien implémentées et ne causent pas de bogues.

b) Votre collègue considère que le switch/case présente un problème de duplication de code et propose le code suivant. Est-ce que vous considérez que sa solution est acceptable et améliore la qualité de votre code ? **Justifiez**

```
1 if(index%2 === 0)
2   this.drawImage(this.originalCanvas.nativeElement,f);
3 if(index == 2)
4   this.drawImage(this.modCnvs.nativeElement,f);
```

a) Voici une liste d'erreurs de qualité de code :

- La conversion forcée de `e.target` et du résultat de l'appel `item(0)` sans vérification.
- La double négation de check lignes 4 et 5.
- La fonction négative isNotValidImage devrait s'appeler isValidImage et retourner si une image est valide. On peut inverser le résultat de l'appel si besoin.
- Le nom de la variable `f` n'est pas descriptif de ce que la variable contient.
- Le nom de la variable `check` n'est pas descriptif non plus.
- L'indentation des case du switch n'est pas faite.
- Les accolades autour de chaque case du switch sont inutiles.

b) Ce nouveau code n'est pas une meilleure solution car :

1. Il ne fonctionne pas pour le cas où **index** est égal à 1.
2. Le fait d'utiliser l'opérateur modulo ici diminue la lisibilité du code.
3. Il faudrait remplacer le double égal par un triple égal pour valider que le type de **index** est vraiment number.

Commentaire :
Très bien !

Question **13**

Correct

Note de 1,00 sur 1,00

L'étape "install" de votre pipeline automatisée sur GitLab exécute la commande **npm ci**. Pourquoi utiliser cette commande plutôt que **npm install** ?

a. Il y a une erreur dans la configuration du pipeline : **npm install** devrait être utilisée plutôt que **npm ci**.

b. **npm ci** veut dire **C**onsole **I**nstall et est simplement un alias de **npm install**.

c. **npm ci** s'assure d'installer les versions exactes des dépendances du projet et toujours reproduire le même environnement, ce qui n'est pas garanti avec **npm install**. ✔

d. Il n'y a pas de différence entre les commandes et les 2 peuvent être utilisées de manière interchangeable.

e. **npm ci** veut dire **C**ontinuous **I**ntegration et doit donc être utilisé dans un processus automatisé.

Votre réponse est correcte.

La réponse correcte est :
npm ci s'assure d'installer les versions exactes des dépendances du projet et toujours reproduire le même environnement, ce qui n'est pas garanti avec **npm install**.

Question **14**

Correct

Note de 1,00 sur 1,00

Une fois qu'une demande de fusion (*Merge Request*) a été revue et approuvée par un membre de l'équipe, qui devrait être responsable de l'intégration du code dans la branche de destination ?

a. Seulement le membre d'équipe ayant implémenté la majorité du code.

b. Le ou les membre(s) designé(s) par une convention quelconque dans l'équipe. ✔

c. Aucune de ses réponses.

d. N'importe quel membre de l'équipe.

e. Seulement un membre de l'équipe qui n'est pas l'auteur ou l'évaluateur de la demande fusion.

f. Seulement le membre d'équipe ayant revisé la majorité du code.

Votre réponse est correcte.

La réponse correcte est :
Le ou les membre(s) designé(s) par une convention quelconque dans l'équipe.

◄ Annonces

Aller à…

Introduction ►