



PLAN TEST

ORINOCO

By FALVO Anthony

Test sur plus de 90% du code javaScript



SCROLL POUR EN SAVOIR PLUS...



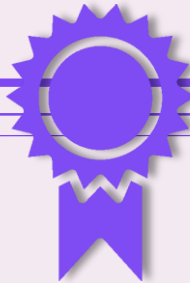


SOMMAIRE :

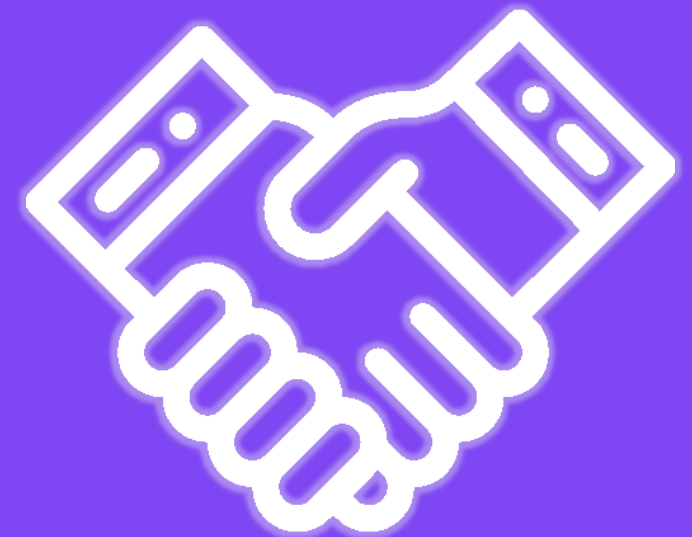
01. PRESENTATION DU PLAN TEST

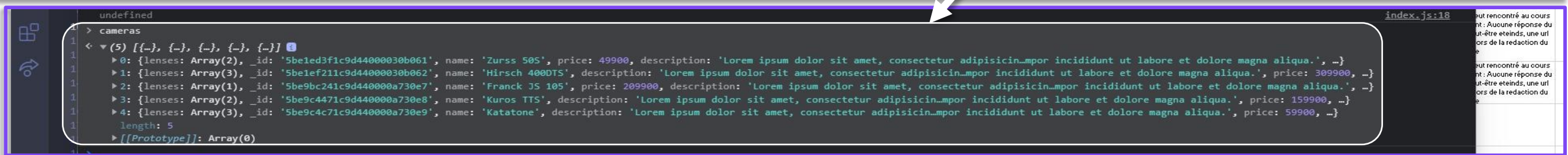

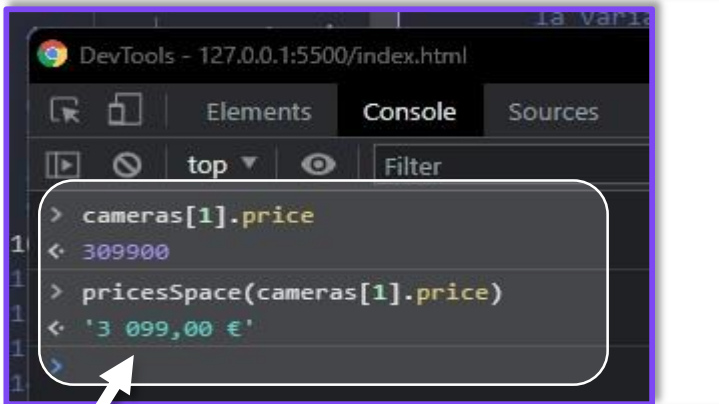


Orinoco

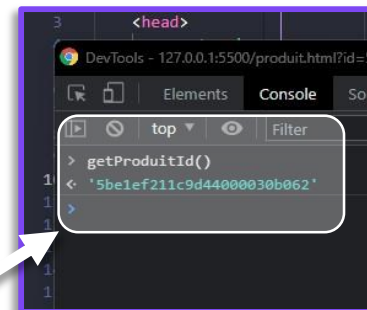


Index.js
produit.js
panier.js
validation.js



Fichier JS	Lignes de code testées	Fonction testée	Résultat attendu	Comment vérifier le résultat attendu	Problème possible
01. index.js	13 à 17	fetchCameras()	La fonction permet de récupérer l'URL de l'api via la methode GET de fetch(). Cette fonction doit nous retourner les informations de ce serveur, afin de les affichés plus tard par l'intermédiaire de la fonction showCameras() que l'on verra en dessous.	Pour vérifier son résultat, le console.log(cameras) doit nous retourner un tableau de 5 éléments. Cameras et variable ou est stocker la methode fetch.	Les problèmes que l'on peut rencontré au cours de cette manipulations sont : Aucune réponse du serveur, le serveur est peut-être eteinds, une url incorrecte, ou un défaut lors de la redaction du code
					
02. index.js	21 à 47	showCameras()	Grâce à la fonction précédente, on attends la réponse de l'API, et l'on insert en HTML les éléments récupérés du tableau.	Pour vérifier son résultat, c'est relativement simple. Si les éléments sont présents sur la page visuellement, c'est que l'appel a bien fonctionner.	Nous pouvons rencontré une erreur dans la console et un non affichage du contenu de l'api.
					Le soucie fréquent lorsque l'on traite des nombres c'est le principe de number et chaine de caractère. On peut ce retrouver avec des valeurs Null et ou NaN not a number. Lors du traitement il est important de vérifier cela.
03. index.js	51 à 57	pricesSpace()	Cette fonction est appelé pour convertir le prix de centime à euros, et retourner via une methode toLocalString pour un affichage visual en euros.	Pour vérifier sont résultat il suffit de taper dans la console : pricesSpace(cameras[1].price) . Si le résultat est bien : '3 099,00 €' alors nous avons réussi	

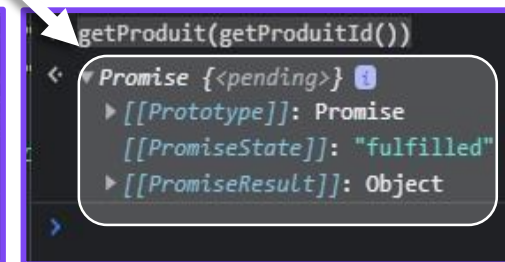
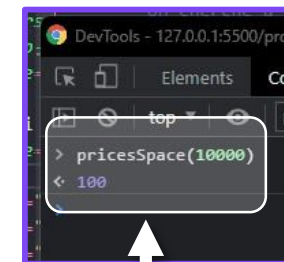
Fichier JS	Lignes de code testées	Fonction testée	Résultat attendu	Comment vérifier le résultat attendu	Problème possible
04. index.js	61 à 85	Fonction callback de buttonShop.addEventListener()	Cette fonction fonctionne au click. Afin d'afficher / masquer, les produits de la boutique	C'est relativement simple dans ce cas, si le comportement n'est pas celui demander. C'est qu'il y a une erreur dans le code.	Les problèmes que l'on peut voir dans cette fonction, c'est des comportement au clique du au changement de style si la fonction getComputedStyle() n'est pas utiliser. Ou simplement un non affichage et ou masquage de la boutique



05. produit.js	20 à 24	getProduitId()	Recupère l'id du produit selectionner au clique sur celui-ci. Et l'ajoute à la fonction suivante que je vous présenterai	Il suffit simplement de taper dans la console le nom de la fonction . Si il retourne un id correspondant au produit. C'est que cela fonctionne.	Le problème peut venir du code, et donc avoir une erreur dans la console.
-------------------	---------	----------------	--	---	---

06. produit.js	28 à 42	getProduit(produitId)	Récupère l'url via fetch() avec comme variable à la fin l'argument defini dans la fonction, et donc par la suite compléter l'url par l'id produit.	il faut taper dans la console getProduit(getProduitId()) si il retourne une promesse avec le State en "fulfilled" c'est que l'appel a fonctionner.	Possible erreur 404, erreur dans l'url. Une erreur s'affichera dans la console grâce au catch()
-------------------	---------	-----------------------	--	--	---




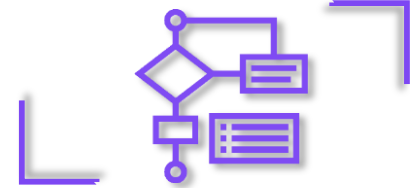
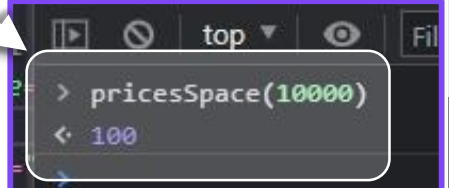
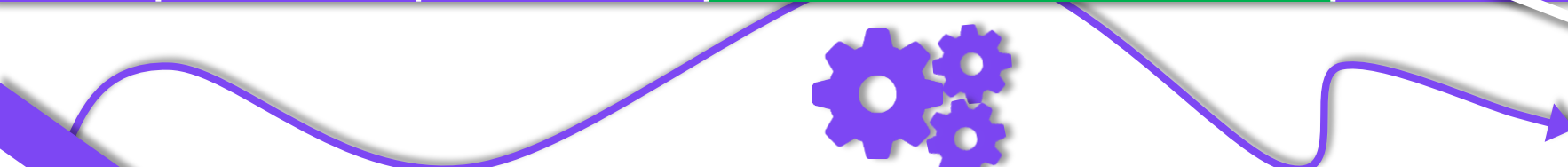
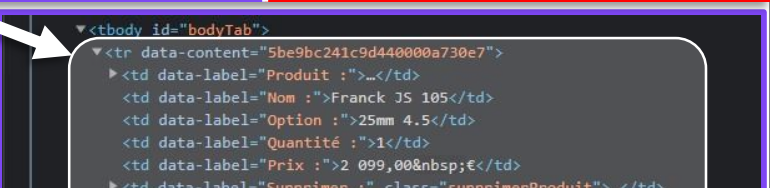
PLAN TEST


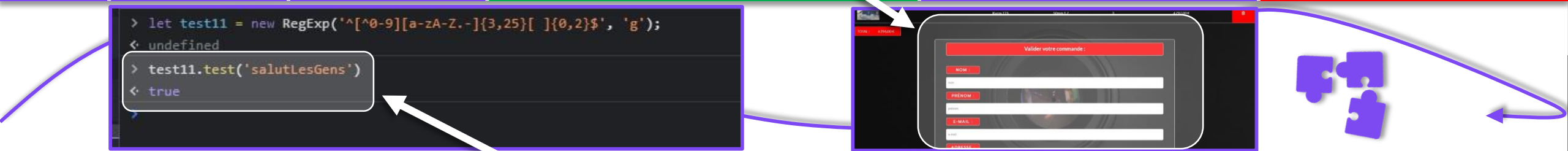
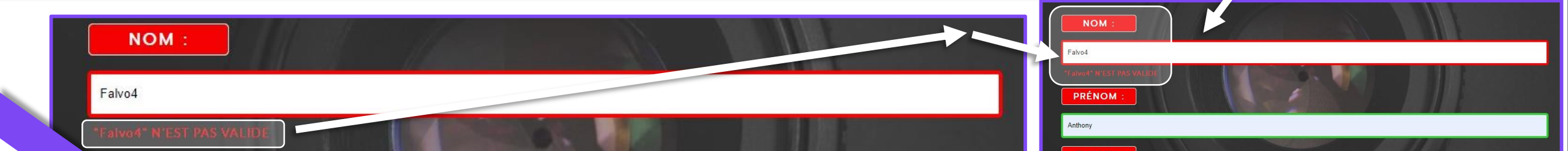


07. produit.js	45 à 49	pricesSpace()	Cette fonction est appelé pour convertir le prix de centime à euros uniquement. Pratiquement identique a la première a l'exception de la methode toLocalString	pricesSpace(100) si le résultat est différent de 1, c'est qu'il y a un problème	identique à la première fonction du même nom. Le soucie number et chaine de caractère. Nous retournera un NaN ou Null
-------------------	---------	---------------	--	---	---

MERCI POUR VOTRE CONFIANCE. LE HIRSCH 400DTS 50MM 1.8 EST AJOUTÉ AU PANIER ☒

AJOUTER AU PANIER

Fichier JS	Lignes de code testées	Fonction testée	Résultat attendu	Comment vérifier le résultat attendu	Problème possible
12. produit.js	112 à 130	optionProduit()	Ajoute chaque option produit lui concernant et l'ajoute en html dans un select pour chaque produit différent de manière dynamique. Si les produits venait a avoir d'autre option, elles seront également prises en comptes.	il suffit de vérifier si chaque produit à toute ces options, et lors de l'ajout au panier que l'option selectionner et la même que celui afficher .	Les erreurs peuvent être simplement une erreur dans la console. De mauvais option a un produit. Ou des produits n'ailant que la première option d'afficher. Il ce peut même que l'option ne soit pas la bonne lors de la mise en panier.
 					
13. produit.js	200 à 255	verificationDesProduits()	Il permet comme decrit dans majProduit() donc de verifier si un element et présent dans le localStorage. Sinon il crée un tableau et y ajoute le produit. Vérifier que le produit et l'option sont les mêmes, si c'est le cas d'y ajouter la quantité. Et si le produit n'est pas déjà présent de l'ajouter. Dans une limite de dix produits différents.	Il suffit de verifier le localStorage et la partie panier. La quantité s'ajoutera si le produit et identique avec l'option. Sinon il le l'ajoutera. Et si aucun tableau n'existe dans le local il en créera un aussi.	Les valeurs en chaines de caractère peuvent posé des problèmes. Les quantités ne pas s'addionner.
14. panier.js	29 à 33	pricesSpace()	Au même titre que les précédentes elle réalise la même chose. Afficher le prix en euros .	Pour la vérifier c'est pareil, un console.log d'une valeur en lançant la fonction et si le resultat est celui attendu alors c'est bon.	identique à la première fonction du même nom. Le soucie number et chaine de caractère. Nous retournera un NaN ou Null
  					
15. panier.js	36 à 475	majPanier()	Un peu comme majProduit() il s'agit d'une fonction regroupant plusieurs petite fonction. Ca fonctionne premier defini une condition . Si panier est vide un message est présent, sinon appel une fonction qui va injecter un tableau. Une autre fonction ajoutera chaque element du local dans ce tableau etc.. Nous detaillerons ca fonction première qui est la condition et la création du talbeau HTML. Le reste serra detaillés ci-dessous.	Pour vérifier il suffit de voir si le tableau et présent dans le code HTML via les devtools ou si le message par defaut du panier vide est bien présent visuellement.	Un message d'erreur est possible dû bien souvent au code. Pour le moment l'appel a l'api ne peut pas présent de problème vu que les produits sont dans le local storage, mais la recuperation de cest element peuvent ne pas s'afficher et retourner une erreur. Donc bien vérifier que la condition est respecté . Que si aucun élément est présent alors le message s'affiche, sinon un tableau en HTML est créer . Le tableau doit aussi bien être afficher sur la partie html des devtools
16. panier.js	126 à 147	ajoutProduitPanier()	Récupère chaque élément du local storage via une boucle et ajoutant une ligne d'un tableau avec c'est différent propriété . Tel que le prix, nom, quantité etc..	Si l'affichage visuel est présente une fois l'article ajouté. C'est que la manipulation fonctionne. On peut aussi le voir sur les devtools.	Le problème qui peu subvenir a ce moment là, est un élément pas présent, et ou une erreur dans la console. Des informations tel que le prix en undefined, ou NaN.
 					

Fichier JS	Lignes de code testées	Fonction testée	Résultat attendu	Comment vérifier le résultat attendu	Problème possible
17. panier.js	74 à 84	totalPanier()	Ajoute le prix de chaque élément du local storage donc du tableau FOIS leur quantité et ajoute ce prix a une variable déclaré en globale. Ainsi ce met a jour a chaque suppression de produit du tableau.	Il suffit d'ajouté des articles au tableau, des quantité, et vérifier le résultat .Si les quantités, et article correspondent bien entre elle.	Alors les soucis peuvent être la non prise en compte des quantités, la non prise en compte des articles, ou un total qui ce met pas a jours a la suppression de l'article. Des soucis comme toujours avec des prix NaN .
					
18. panier.js	154 à 173	deleteProduit()	Supprime l'article via une icone	Une fois cliquer sur l'icone. L'élément doit être retirer du localStorage et visuellement.	Problème possible : Suppression du local mais pas visuel, suppression visuel mais pas du local. Ou les deux avec une erreur.
19. panier.js	186 à 230	inserFormulaire()	insert le formulaire lorsque le client est prêt à commander via un bouton créer précédemment.	Au clique si le formulaire s'affiche c'est que c'est fonctionnelle. On peut aussi voir dans les devtools si le html a bien était créer	Non affichage du formulaire est erreur dans la console.
					
20. panier.js	312 à 365	validationLastName() validationFirstName() ValidationEmail() validationAddress() validationCity()	Les fonctions font la même chose, sauf qu'ils possèdent des regexp bien différent. On attend d'elle qu'elle vérifie la validité du formulaire, via les inputs. Et renverra TRUE ou FALSE en fonction de si les conditions sont respecté ou non.	Un console.log permet de testé la regexp, certain site spécialisé le font aussi. Vous pouvez testé directement la condition sur la console, elle renverra un booléen .	La condition renvoie toujours vrai, ou toujours faux. Ou une erreur dans la console .
21. panier.js	273 à 307	conditionValidation()	condition permettant de verifier le true et false des regexp, et affiche un message d'erreur si false . Si true affiche une bordure verte. Si rien est indiquer, il garde sont comportement pas default.	il suffit d'entrer des informations et voir si visuellement le comportement correspond au valeur indiquer dans la console par les regexp. Qu'ils prennent bien en compte le true / false	Qu'il ne réagisse pas au vrai / faux, une erreur dans la console. Ou qu'il n'affiche simplement pas la partie visuel Rouge / Vert ou par default.
					



FIN



J'espère que ce plan test vous a plus. J'y est passé beaucoup beaucoup de temps pour ça réalisation. Afin de faire quelque chose de simple et original.



La vie est une formation continue ...



Les sources de ce diapositive / pdf :

les icônes sur flavicon, l'image du sommaire sur openclassrooms ainsi que le logo OC. Le reste est entièrement fait à la main avec la fonction Drawing de powerpoint et Gimp. Les polices sont du lato, lato black, josefin sans. Les captures sous windows 11 bêta.