

TESTER L'ACCESSIBILITÉ

Enjeux et pratiques pour le développement Web & Mobile de SNCF Connect







- Intro : Nous, l'accessibilité et pourquoi la tester, SNCF Connect
- Etudes de cas & solutions techniques
 Tests autos, tests manuels
- Vision organisationnelle









eSNCF SOLUTIONS

Nicolas Chardon



Adrien Body

#EXPERT
#DEV #WEB

#EXPERT #A11Y #EXPERT
#DEV #MOBILE



"The power of the Web is in its universality.

Access by everyone regardless of disability is an essential aspect. »

Tim Berners-Lee, W3C Director and inventor of the World Wide Web



ACCESSIBILITÉ





- Perceptible
- Utilisable
- Compréhensible
- Robuste

ACCESSIBILITÉ



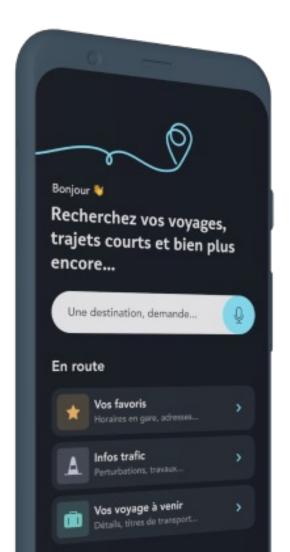


Pourquoi tester l'Accessibilité?

- 12 millions de personnes en France ont un handicap + 15 millions vont développer des handicaps (âge, maladie)
- La loi française (service public)
- Bonus : faciliter les usages en général

SNCF CONNECT







- Site Web & App Mobile (iOS/Android)
- +3 millions de visites quotidiennes
- Tech : React & Flutter
- Equipe : 300+ contributeurs développeurs et tech





ETUDES DE CAS

Cas d'usage

- → solutions automatisables (Web / Mob.)
- → tests manuels complémentaires



Je suis un utilisateur malvoyant...

...Je dois pouvoir augmenter la taille du texte







(Dev #Web avec React)

Utilisation de l'unité relative : rem

```
export const style = css({
  fontSize: rem(24), // 24px, 1.5rem
})
```

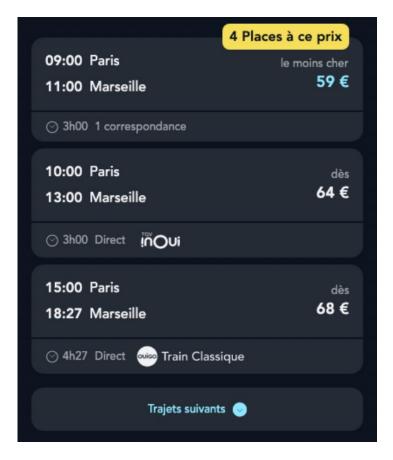
Carte Avantage Adulte

Avec la carte Avantage, bénéficiez de prix plafonnés en 2nde classe sur TGV INOUI et INTERCITÉS en direct, en dernière minute et même en période de grands départs!

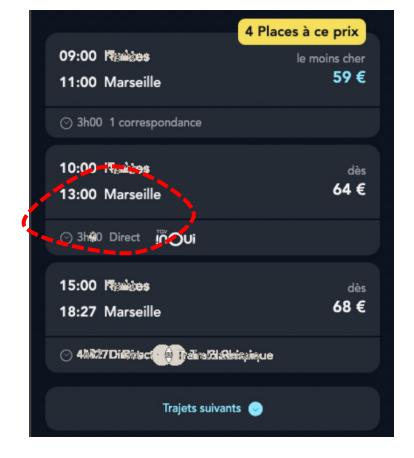
- Pour vos trajets de moins d'1h30 = 39€ max
- Pour vos trajets entre 1h30 et 3h = 59€ max
- Pour vos trajets de plus de 3h = 79€ max

(Dev #Mobile avec ← Flutter)

Utilisation de tests Golden



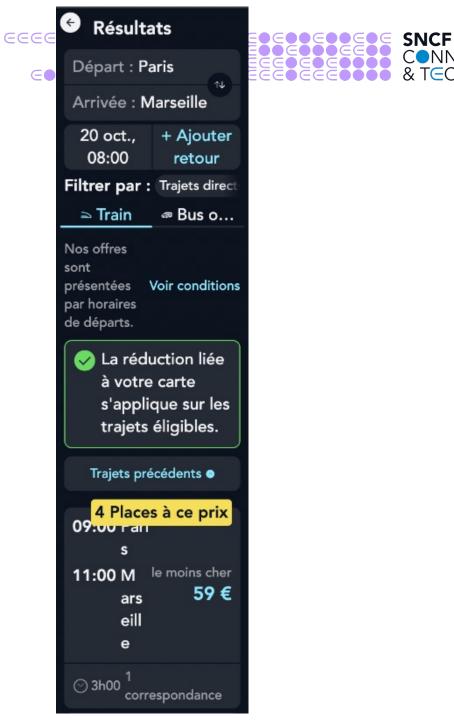
Référence



Différence superposée

(Dev #Mobile avec ← Flutter)

Des tests *Golden* version a11y :





(Dev #Mobile avec < Flutter)

```
testWidgets('Golden test', (WidgetTester tester) async {
 await tester.pumpWidget(MyApp());
 await expectLater(find.byType(MyApp),
matchesGoldenFile('main.png'));
});
static final a11yScreen = Device.phone.copyWith(name: 'a11y',
textScale: 2);
// Golden Toolkit
multiScreenGolden
```



(Tests manuels #QA 🐠)

- Tests manuels Web :
 - Augmenter la taille du texte à 200% via Zoom navigateur
 - Augmenter la taille du texte à 200% via Zoom texte du navigateur (Firefox seulement)
- Tests manuels Mobile → paramètres systèmes
 - Agrandir la taille des boutons
 - Agrandir la taille du texte à 200%

Référence WCAG : Success Criterion 1.4.4: Resize Text (w3.org)

- Vérifier :
 - Absence de chevauchements de textes
 - Absence de textes tronqués
 - Les boutons & textes s'agrandissent fonction des paramètres (applications mobiles)



...J'ai besoin d'indications sémantiques pour naviguer avec des outils spécialisés

SÉMANTIQUE



(Tests manuels #QA 🕦)

- Structure HTML pertinente : <header>, <nav>, <main>, <footer>
- Structure de titres pertinente : <h1>, <h2>, <h3>, <h4>, <h5>, <h6>
- Pour vérifier :
 - Extension Chrome <u>Landmark Navigation</u>
 - Extension Chrome <u>HeadingsMap</u>





SÉMANTIQUE



(Dev #Web avec React)

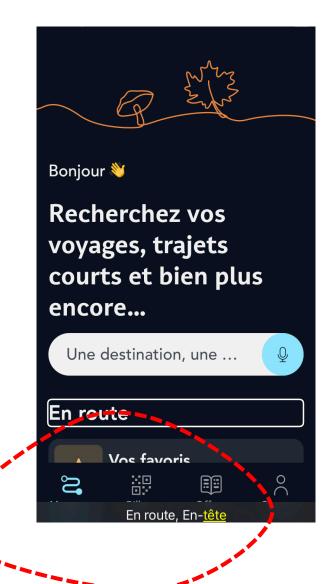
Linter ESLint avec eslint-plugin-jsx-a11y :

```
114
             <div css={styles.actions(theme)}>
115
               <button aria-selected />
116
                (property) JSX. Intrinsic Elements. button: Detailed HTMLProps < Button HTMLAttributes < HTMLButton Element>,
117
                HTMLButtonElement>
118
                Missing an explicit type attribute for button eslint(react/button-has-type)
119
120
                The attribute aria-selected is not supported by the role button. This role is implicit on the element
121
                button. eslint(jsx-a11y/role-supports-aria-props)
122
                Voir le problème (F2) Correction rapide... (∑Enter)
123
       export
```

SÉMANTIQUE

(Tests manuels #QA ()

- Besoin de titres dans les écrans qui délimitent les zones de contenu
- Vérification manuelle : parcourir les titres de l'écran à l'aide du lecteur d'écran



```
(Dev #Mobile avec < Flutter)
```

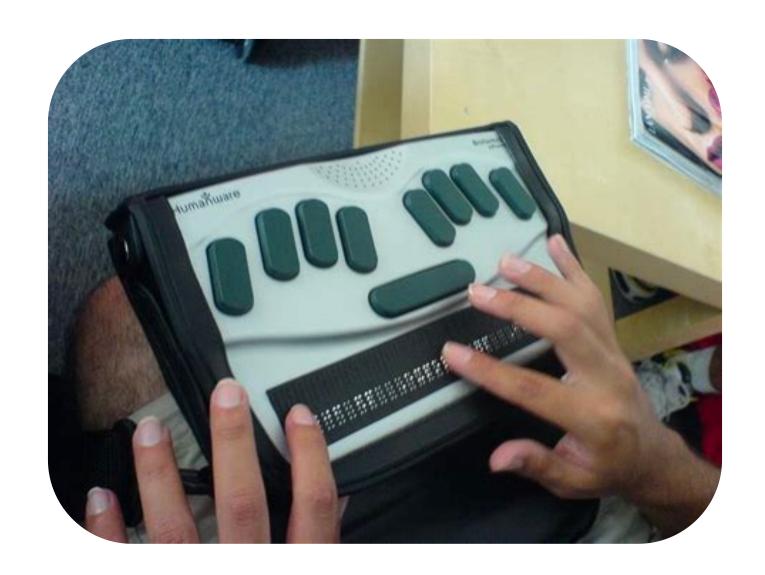
On peut tester qu'un composant dans un test widget a la bonne sémantique pour le lecteur d'écran :

```
expect(tester.getSemantics(find.byType(Semantics)),
matchesSemantics(
    hasTapAction: true,
    onTapHint: 'test',
    isTitle: true
));
```



...J'ai besoin des étiquettes textuelles sur les éléments porteurs d'information

(Illustration / Explication)





(Dev #Web avec React)

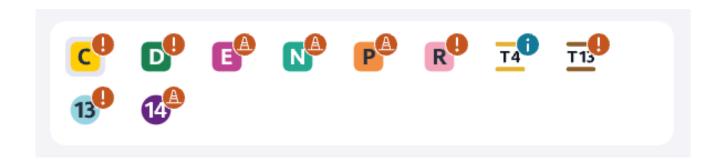
Tests composants Web avec :



« The more your tests resemble the way your software is used, the more confidence they can give you. » — Kent C. Dodds

Sélecteurs getByRole, getByLabelText...

(Dev #Web avec React)



```
test('TrafficInfo affiche bien les lignes perturbées', () => {
   render(<TrafficInfo />)
   expect(
    screen.getByRole('button', { name: /RER D, Trafic perturbé/i })
   ).toBeInTheDocument()
})
```

(Dev #Mobile avec ← Flutter)

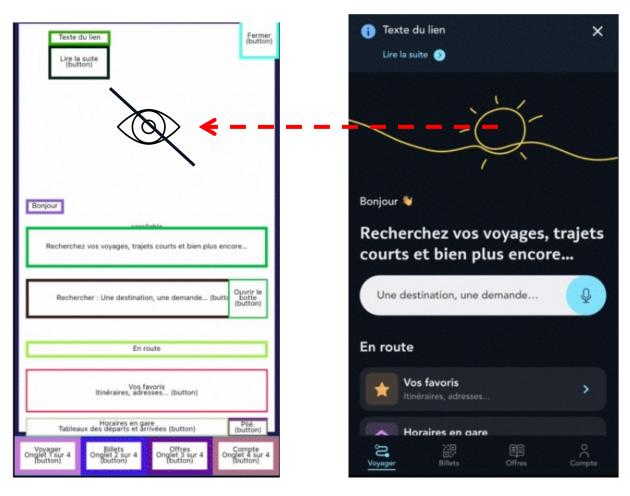
Tests unitaires automatisés sur les labels

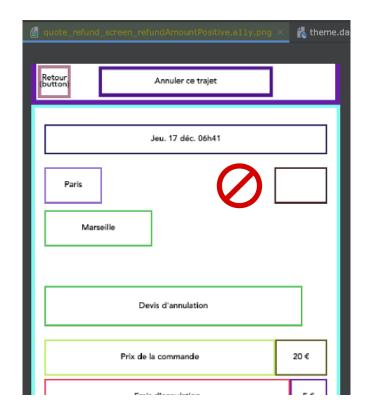
```
testWidgets('Button label is valid', (WidgetTester tester) async {
  await tester.pumpWidget(NiceAccessibleButton());
  expectLater(tester, meetsGuideline(labeledTapTargetGuideline));
})
```

(Dev #Mobile avec ← Flutter)

Golden tests & tests manuels en mode accessibilité

(enableSemanticDebugger)





(Dev #Mobile avec ← Flutter)

Tests Widgets

```
testWidgets('Expect first tab title to be have the infoTrafficRegion label',
(WidgetTester tester) async {
await tester.pumpWidgetBuilder(TrafficInfoScreen(), textScaleSize: 2);
final finder = find.byWidgetPredicate(
       (widget) =>
  widget is Text &&
       widget.semanticsLabel == I18n.current.infoTrafficRegion title,
);
expect(finder, findsOneWidget);
});
```



(Tests manuels #QA 🕦)

Outils de développements du navigateur : inspecteur d'accessibilité



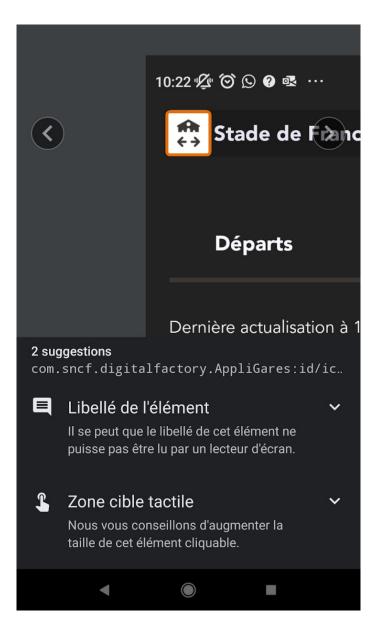


Vérifications automatiques (présence de label) et semi automatique (pertinence du label) : axe-core et outils basés dessus.

(Tests manuels #QA 🐠)

- Tests avec lecteur d'écran (Voice Over iOS ou Android)
- Tests semi-auto
 (Accessibility Scanner Android)



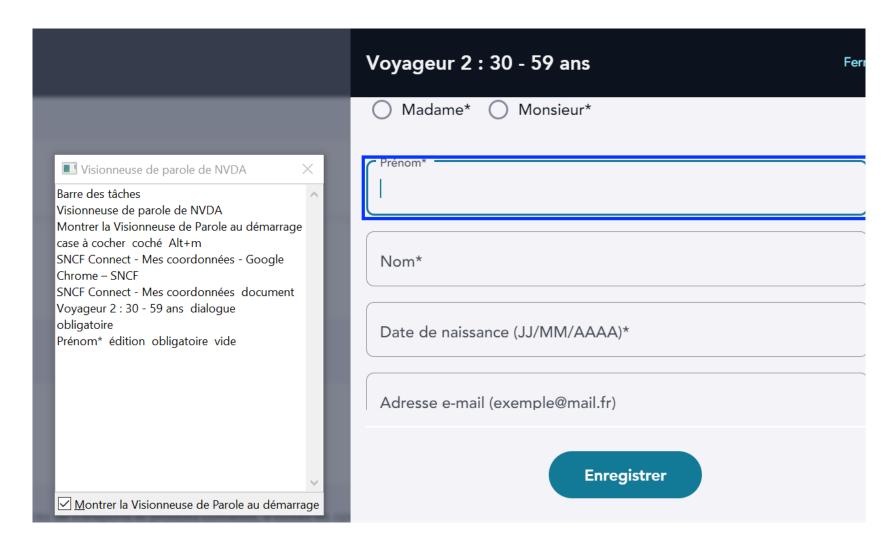




...Je dois pouvoir remplir et utiliser un **formulaire** avec un **lecteur** d'écran



(Illustration / Explication)

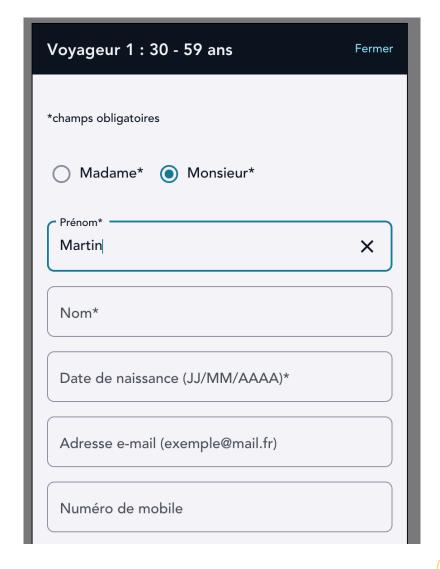


(Dev #Web avec React)



```
await user.type(
   screen.getByLabelText(/Prénom/i),
   'Martin'
)
```







(Dev #Mobile avec ← Flutter)

- Tests Widgets ou Integration Tests avec ciblage par label comme en React
- TUs sur les labels avec labeledTapTargetGuideline

```
final widget = finder.bySemanticsLabel("Groupe de Cartes Déplié.");
await tap(widget);
```

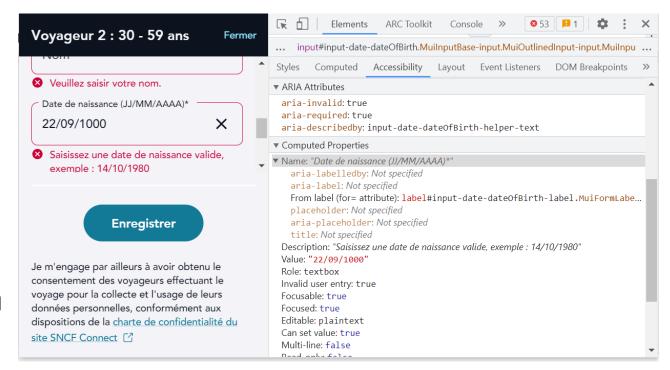




- Étiquette pertinente, sémantique, états
- Indication des champs obligatoires

 ("* obligatoire" en début du formulaire)
- Liaisons avec les messages d'erreur
- Formats attendus (numéros de téléphone, code postal, etc).

Outils : outils de développement de Chrome ou Firefox





(Tests manuels #QA 🕦)

Tests lecteurs d'écran mobile (VoiceOver, Talkback) (cf. Liens en fin de présentation)

Fournir un retour vocal sur une action utilisateur

- erreur de saisie dans le formulaire empêchant l'envoi
- erreur réseau/ succès

ANNONCES SEMANTIQUES



(Dev #Mobile avec ← Flutter)

On peut tester via des mocks le texte envoyé au Semantic Service Flutter. Ce service est appelé dans le cas où le Semantics.liveRegion ne suffit pas

```
testWidgets('Announce semantic selected on radio switch button pressed',
(WidgetTester tester) async {
 final mock = MockSemanticAnnouncements(tester);
 const expectedAnnouncement = AnnounceSemanticsEvent('sélectionné, Choix
1', TextDirection.ltr);
 const widget = RadioSwitchButtons();
 await tester.pumpWidgetBuilder(widget);
 final button = find.byType(Text).first;
 await tester.tap(button);
 await tester.pumpAndSettle();
 expect(mock.announcements, hasOneAnnouncement(expectedAnnouncement));
});
```



...Je dois dois avoir des **Contrastes** conforme

CONTRASTE



(Dev #Mobile avec ← Flutter)

TU sur le contraste d'un widget à faire en thème sombre et thème clair !

meetsGuideline(textContrastGuideline)



...J'ai besoin d'une interface facilement utilisable

CECECO CONNECT CONNECT & TECH

(Dev #Mobile avec ← Flutter)

TUs Automatiques

meetsGuideline(androidTapTargetGuideline) meetsGuideline(iOSTapTargetGuideline)

ZONE DE TAP



(Tests manuels #QA)

Référence : <u>Understanding Success Criterion 2.5.8: Target Size (Minimum) (w3.org)</u>

- Accessibility Scanner sur Android
- Vérification manuelle : taille minimale de 24px sur 24px CSS (nouvelles exigences AA de WCAG 2.2)



...J'ai besoin utiliser mon app avec différentes tailles d'affichage

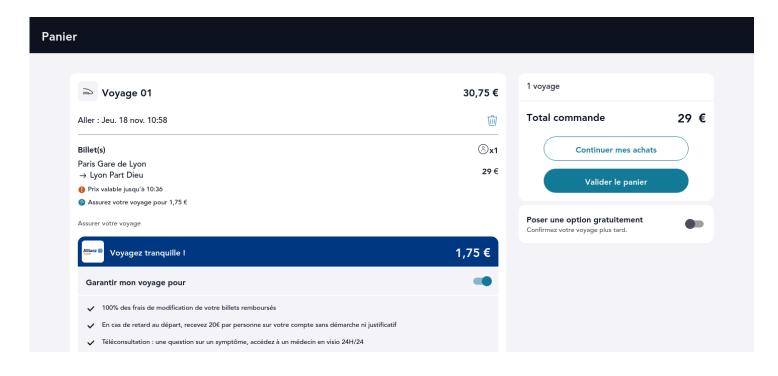
MODE D'AFFICHAGE



(Dev #Web avec React)

- Affichage des composants en isolation avec S Storybook
- Screenshots Web desktop & Web mobile pour les composants page

Panier	
1 voyage	
Total commande	29 €
Poser une option gratuiteme Confirmez votre voyage plus tard.	nt •
⇒ Voyage 01	30,75 €
Aller : Jeu. 18 nov. 10:58	Ŵ
Billet(s) Paris Gare de Lyon	⊗x1
. a.i. cai c as Lyon	22.0





...J'ai besoin de pouvoir l'orientation de mon app

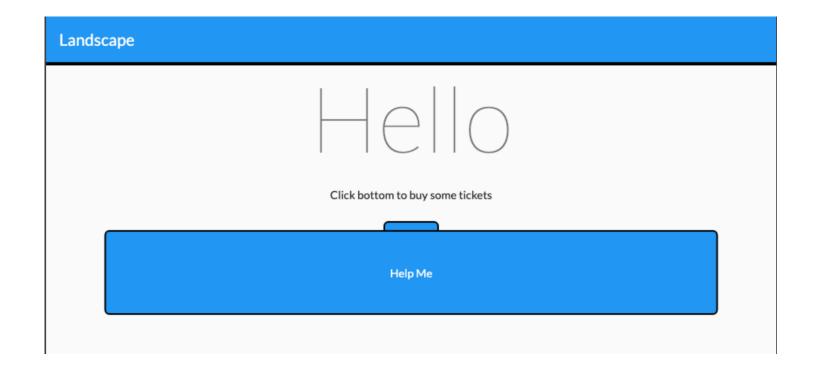
MODE D'AFFICHAGE

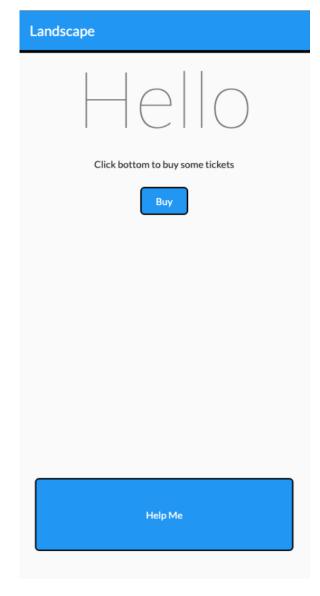
CECCEC CONNECTOR & TECH

(Dev #Mobile avec ← Flutter)

Goldens Landscape Mode







MODE D'AFFICHAGE



(Tests manuels #QA 🐠)

- Vérifier qu'il n'y a pas de chevauchements de textes
- Vérifier l'absence de textes tronqués



...J'ai besoin de naviguer avec un **Clavier**

NAVIGATION CLAVIER



(Tests manuels #QA 🕦)

- Vérifier la présence d'un style visuel de prise de focus (natif ou personnalisé)
- Astuce devtools : expression espion "document.activeElement" pour identifier sur quel élément le focus est positionné.
- Vérifier que l'ordre de tabulation respecte l'ordre visuel : <u>extension Chrome Taba11y</u>
- Vérifier que tous les éléments interactifs peuvent être activés avec le clavier : boutons, cases à cocher, boutons radio...

NAVIGATION CLAVIER



(Tests manuels #QA 🕦)

- Connecter un clavier externe (bluetooth ou USB)
- Activer "Accès complet au clavier" (iOS) ou Switch Access (Android)
- Vérifier que tous les éléments interactifs peuvent être la cible du focus clavier
- Vérifier la présence d'un style visuel de focus clavier sur tous les éléments interactifs
- Vérifier la possibilité d'interagir au clavier avec ces éléments



RÉCAPITULATIF...

Beaucoup de situations en prendre en compte

Beaucoup de solutions techniques → tests composants, linters, ESLint, tests de régression visuelle, tests Golden, tests Widgets, tests unitaires meetsGuideline...

...Et puis c'est tout ?





Accessibilité & organisation, retour d'expérience de Voyages-sncf.com à SNCF

Connect





En 2018 : création d'un poste de référent technique accessibilité

- Suivi des audits
- Sensibilisation et formations
- Accompagnement de la conception
- Accompagnement des développements
- Accompagnement des tests manuels
- Travail en collaboration avec les utilisateurs handicapés : associations représentatives, consultant malvoyant





- Basé sur des référentiels : RGAA 4.1, RAAM1
- Sur un échantillon représentatif des usages et/ou des contenus
- Vérifie les critères applicables sur chaque page/écran
- Un critère KO sur une page = KO sur tout l'échantillon
- Taux de conformité légale = nombre de critères OK / nombre de critères applicables
- Conformité légale Web en janvier 2022 : 54% -> 70% en juillet 2022 (13 pages)
- Conformité moyenne App en janvier 2022 : 67% (13 écrans + modales)



Liberté Égalité Fraternité

RÉFÉRENTIEL GÉNÉRAL D'AMÉLIORATION DE L'ACCESSIBILITÉ



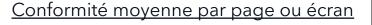
RAAM1

Référentiel d'évaluation de l'Accessibilité des Applications Mobiles



DES EFFORTS PAYANTS





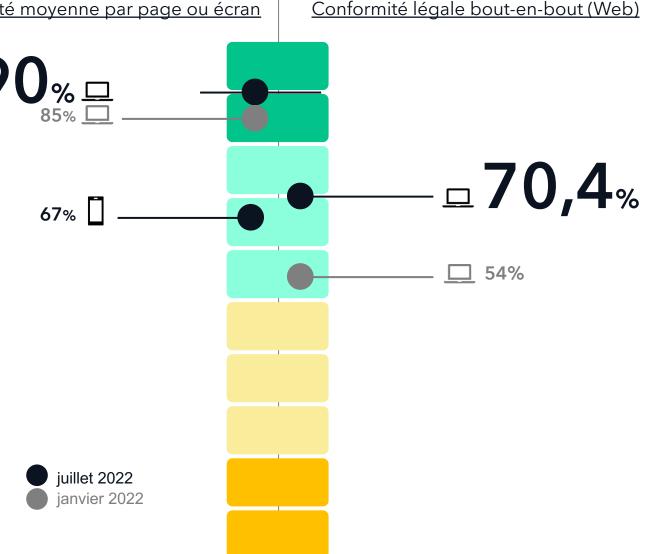
Conformité légale bout-en-bout (Web)

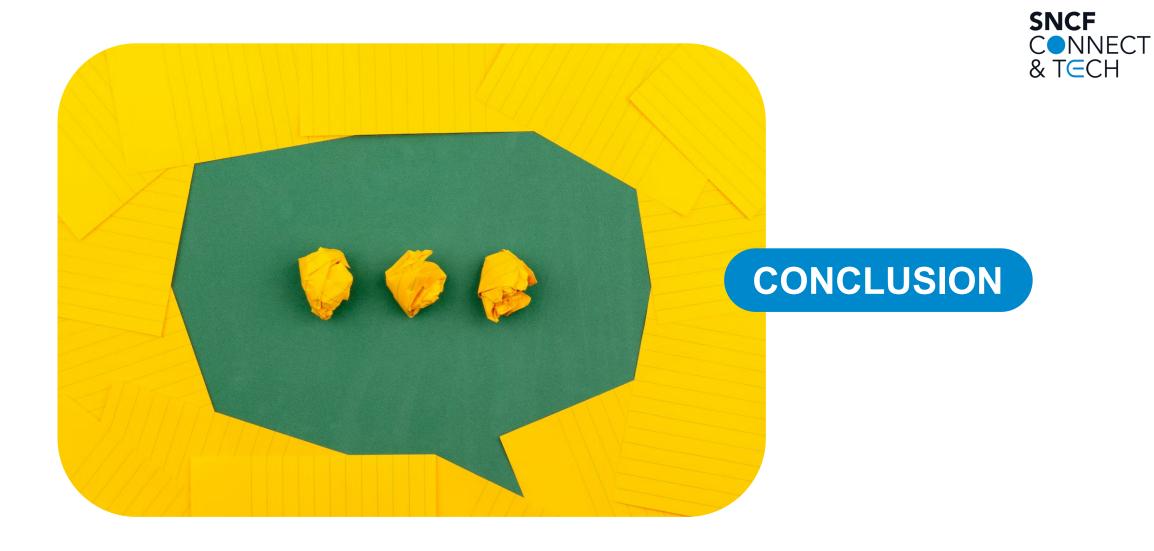
Un dispositif qui fait ses preuves...

Un taux de conformité en forte croissance, avec un <u>objectif porté à 75% sans blocage</u> sur 2023.

... mais qui doit être renforcé pour améliorer notre conformité légale

Miser sur l'industrialisation des bonnes pratiques de conception et développement.











- L'accessibilité, trivial ni à implémenter, ni à tester!
- Il y a des solutions : organisation & techniques (Web, Mobile)
- Toujours des choses à améliorer







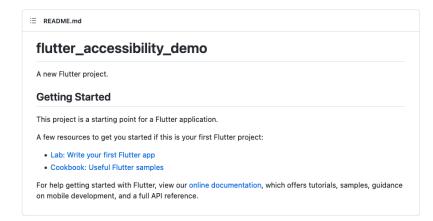
https://jobs.connect-tech.sncf/



RÉFÉRENCES



https://www.youtube.com/watch?v=kuW-uvywcWo







Accessibility in Flutter: Tips for a more inclusive app

When you build an application, you must think how people with disabilities will be able to use it. But to get a fully accessible app is not that easy. In this article I will try to help with some tips or practices me and my team discovered during our journey with Flutter. Some are automated, some are more manual, in any case I will try to show you tips that are not discussed in the official accessibility documentation.

https://abody.medium.com/ accessibility-in-flutter-tips-for-a-more-inclusive-app-3e8406e9356d

RÉFÉRENCES



- https://developer.apple.com/documentation/uikit/accessibility for uikit
- https://developer.android.com/guide/topics/ui/accessibility/
- https://www.bbc.co.uk/accessibility/forproducts/guides/mobile/
- https://a11y-guidelines.orange.com/en/mobile/android/
- https://a11y-guidelines.orange.com/en/mobile/ios/
- https://flutter.dev/docs/development/accessibility-and-localization/accessibility
- https://flutter.dev/docs/development/ui/widgets/accessibility
- https://medium.com/flutter-community/a-deep-dive-into-flutters-accessibility-widgets-eb0ef9455bc
- https://www.youtube.com/watch?v=bWbBgbmAdQs
- https://www.youtube.com/watch?v=kuW-uvywcWo
- https://github.com/voyages-sncf-technologies/flutter_accessibility_demo
- https://abody.medium.com/accessibility-in-flutter-tips-for-a-more-inclusive-app-3e8406e9356d