

comme mettre en place un mécanisme permettant de demander à l'utilisateur d'enregistrer ses modifications avant de quitter une page ?

ChatGPT said:

Pour mettre en place un mécanisme permettant de demander à l'utilisateur d'enregistrer ses modifications avant de quitter une page web (par exemple s'il modifie un formulaire), on utilise généralement l'événement **beforeunload** du navigateur.

Voici comment procéder :



Exemple de base en JavaScript :

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Formulaire</title>
</head>
<body>
  <form id="monFormulaire">
    <label>Nom :
      <input type="text" name="nom">
    </label>
    <br>
    <label>Prénom :
      <input type="text" name="prenom">
    </label>
    <br>
    <button type="submit">Enregistrer</button>
  </form>

  <script>
    let isModified = false;

    // Détection de modification dans le formulaire
    const form = document.getElementById('monFormulaire');
    form.addEventListener('input', () => {
      isModified = true;
    });

    // Réinitialisation après soumission
    form.addEventListener('submit', () => {
      isModified = false;
    });

    // Avant de quitter la page
    window.addEventListener('beforeunload', (event) => {
      if (isModified) {
        event.preventDefault();
        // Message personnalisé non pris en compte par tous les navigateurs
        event.returnValue = ''; // Nécessaire pour certains navigateurs comme
        Chrome
      }
    });
```

```
</script>
</body>
</html>
```

Explication :

- `isModified` : une variable qui passe à `true` dès que l'utilisateur modifie le formulaire.
 - L'événement `beforeunload` s'active lorsqu'un utilisateur tente de quitter la page (fermer l'onglet, recharger, cliquer sur un lien...).
 - Le navigateur affichera un **message de confirmation automatique** (non personnalisable dans la plupart des navigateurs modernes pour des raisons de sécurité).
-

Limitations à connaître

- Le **message de confirmation n'est pas personnalisable** dans la majorité des navigateurs récents (ex : Chrome, Firefox, Edge). Le navigateur affichera son propre message générique.
 - Ce mécanisme peut être bloqué si l'utilisateur a désactivé les alertes de navigation.
 - N'abusez pas de cette fonctionnalité : elle ne doit être utilisée **que si nécessaire**, sinon l'expérience utilisateur en souffre.
-

Astuce UX :

Pour une meilleure expérience utilisateur :

- Ajoutez aussi un **indicateur visuel** (ex : "Modifications non enregistrées").
- Sauvegardez automatiquement les modifications dans le stockage local (`localStorage`) en cas de départ inattendu.
- Ajoutez un bouton "Annuler les modifications" pour plus de contrôle.

Souhaitez-vous un exemple avec sauvegarde automatique ou stockage temporaire ?

You said:

je souhaite personnaliser le message

ChatGPT said:

C'est tout à fait compréhensible, mais malheureusement **les navigateurs modernes ne permettent plus de personnaliser le message** affiché dans la boîte de dialogue du `beforeunload` pour des raisons de **sécurité et d'expérience utilisateur**.

❌ Pourquoi la personnalisation est bloquée ?

Historiquement, les développeurs pouvaient écrire un message personnalisé, comme :

```
event.returnValue = "Vous avez des modifications non enregistrées. Voulez-vous vraiment quitter ?";
```

Mais aujourd'hui, **la plupart des navigateurs ignorent complètement le texte personnalisé** et affichent à la place un message standard, du type :

"Des modifications que vous avez apportées peuvent ne pas être enregistrées."

Cela s'applique à :

- Chrome
- Firefox
- Safari
- Edge



Que peut-on faire alors ?

Même si on ne peut pas personnaliser **la boîte de dialogue du navigateur**, voici des alternatives pour **mieux gérer l'expérience utilisateur** :



1. Créer un dialogue de confirmation personnalisé à l'intérieur de votre application

Utilisez **un système de routage ou de détection d'action utilisateur**, et affichez un **modal (popup custom)** si l'utilisateur clique sur un lien ou un bouton susceptible de le faire quitter la page.

Exemple avec un lien interne :

```
<a href="autre-page.html" id="quitterPage">Aller vers une autre page</a>

<script>
  let isModified = false;

  document.querySelector('form').addEventListener('input', () => {
    isModified = true;
  });

  document.getElementById('quitterPage').addEventListener('click', (e) => {
    if (isModified) {
      const confirmation = confirm("Vous avez des modifications non enregistrées. Voulez-vous vraiment quitter ?");
      if (!confirmation) {
        e.preventDefault(); // Bloque la navigation
      }
    }
  })
</script>
```

```
}  
});  
</script>
```

✅ Ici, le message est totalement personnalisable.

🧠 2. Dans les Single Page Applications (React, Vue, Angular, etc.)

Les frameworks comme **React Router**, **Vue Router**, etc., permettent d'intercepter les changements de route. Vous pouvez y intégrer vos propres dialogues personnalisés (modals ou `window.confirm`).

📌 Résumé

Méthode	Message personnalisable ?	Remarques
<code>beforeunload</code>	❌ Non	Utilisé pour les fermetures ou rechargements
<code>window.confirm()</code> custom	✅ Oui	Pour les clics sur des liens ou boutons internes
Modals personnalisés (HTML/CSS/JS)	✅ Oui	Pour une UX plus élégante
Frameworks SPA (React, Vue...)	✅ Oui	Pour des applis plus complexes

Souhaitez-vous que je vous montre comment faire un **modal personnalisé en JavaScript pur**, ou que je vous aide à l'intégrer dans un framework comme React ou Vue ?

You said:

On a une solution pour les onglets type bootstrap ?

ChatGPT said:

Oui, **si tu utilises des onglets Bootstrap** (par exemple avec le composant `nav-tabs` ou `tab-pane`), tu peux **intercepter le changement d'onglet** pour afficher un **message personnalisé** si l'utilisateur a des modifications non enregistrées **avant de quitter un onglet actif**.

👉 **Bonne nouvelle** : comme il ne s'agit pas d'un rechargement de page, tu peux **afficher un message 100% personnalisé**, via `window.confirm()` ou un **modal Bootstrap**.

✅ Exemple complet avec Bootstrap (JavaScript pur)

🎯 Objectif :

- Bloquer le changement d'onglet si des modifications ont été faites.

- Afficher une confirmation personnalisée.
 - Autoriser le changement si l'utilisateur confirme.
-

HTML de base (avec onglets Bootstrap)

```
<!-- Onglets -->
<ul class="nav nav-tabs" id="monTab" role="tablist">
  <li class="nav-item">
    <a class="nav-link active" data-bs-toggle="tab" href="#onglet1">Onglet 1</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" data-bs-toggle="tab" href="#onglet2">Onglet 2</a>
  </li>
</ul>

<!-- Contenu des onglets -->
<div class="tab-content">
  <div class="tab-pane fade show active" id="onglet1">
    <form id="form1">
      <input type="text" name="nom" placeholder="Nom">
    </form>
  </div>
  <div class="tab-pane fade" id="onglet2">
    <p>Contenu de l'onglet 2</p>
  </div>
</div>
```

JavaScript pour intercepter le changement d'onglet

```
<script>
  let isModified = false;

  // Détection de modification dans le formulaire
  document.querySelector('#form1').addEventListener('input', () => {
    isModified = true;
  });

  // Interception du changement d'onglet
  const tabLinks = document.querySelectorAll('a[data-bs-toggle="tab"]');
  tabLinks.forEach(link => {
    link.addEventListener('click', function (e) {
      if (isModified) {
        const confirmed = confirm("Vous avez des modifications non enregistrées.  
Voulez-vous quitter cet onglet ?");
        if (!confirmed) {
          e.preventDefault(); // Empêche le changement d'onglet
        } else {
          isModified = false; // Réinitialiser si l'utilisateur confirme
        }
      }
    });
  });
</script>
```

Résultat :

- Si l'utilisateur modifie le champ dans **Onglet 1**, puis clique sur **Onglet 2**, une alerte personnalisée s'affiche.
 - S'il accepte, l'onglet change. Sinon, il reste sur l'onglet actif.
-

Aller plus loin :

Tu peux remplacer `confirm()` par un **modal Bootstrap** pour une UX plus élégante.

Souhaites-tu que je te montre comment faire cette version avec un **modal Bootstrap custom** ?

You said:

oui

ChatGPT said:

Parfait ! Voici comment faire une **confirmation de changement d'onglet avec un modal Bootstrap personnalisé**, pour remplacer le classique `window.confirm()`.

Fonctionnement recherché :

- Si l'utilisateur a modifié un formulaire (par exemple dans un onglet),
 - et qu'il tente de **changer d'onglet**,
 - un **modal Bootstrap personnalisé** s'affiche pour lui demander de confirmer ou annuler,
 - si confirmé → l'onglet change,
 - si annulé → il reste sur l'onglet actuel.
-

Étapes :

1. HTML des onglets + formulaire
 2. Modal Bootstrap custom
 3. JavaScript pour détecter les modifications + gérer le changement d'onglet
-



1. HTML – Onglets + Contenu

```

<!-- Onglets Bootstrap -->
<ul class="nav nav-tabs" id="monTab" role="tablist">
  <li class="nav-item">
    <a class="nav-link active" data-bs-toggle="tab" href="#onglet1">Onglet 1</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" data-bs-toggle="tab" href="#onglet2">Onglet 2</a>
  </li>
</ul>

<!-- Contenu des onglets -->
<div class="tab-content mt-3">
  <div class="tab-pane fade show active" id="onglet1">
    <form id="form1">
      <input type="text" name="nom" class="form-control" placeholder="Nom">
    </form>
  </div>
  <div class="tab-pane fade" id="onglet2">
    <p>Contenu de l'onglet 2</p>
  </div>
</div>

```



2. Modal Bootstrap personnalisé

```

<!-- Modal de confirmation -->
<div class="modal fade" id="confirmModal" tabindex="-1" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Modifications non enregistrées</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Fermer"></button>
      </div>
      <div class="modal-body">
        <p>Vous avez des modifications non enregistrées. Voulez-vous vraiment changer d'onglet ?</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Annuler</button>
        <button type="button" class="btn btn-primary" id="confirmerChangement">Oui, continuer</button>
      </div>
    </div>
  </div>
</div>

```



3. JavaScript

```

<script>

```

```

let isModified = false;
let nextTabLink = null;

// Bootstrap modal instance
const confirmModal = new
bootstrap.Modal(document.getElementById('confirmModal'));

// Détection de modification du formulaire
document.querySelector('#form1').addEventListener('input', () => {
  isModified = true;
});

// Intercepter les clics sur les onglets
const tabLinks = document.querySelectorAll('a[data-bs-toggle="tab"]');
tabLinks.forEach(link => {
  link.addEventListener('click', function (e) {
    if (isModified) {
      e.preventDefault();
      nextTabLink = link;
      confirmModal.show(); // Affiche le modal
    }
  });
});

// Si l'utilisateur confirme dans le modal
document.getElementById('confirmerChangement').addEventListener('click', () => {
  if (nextTabLink) {
    const tab = new bootstrap.Tab(nextTabLink);
    tab.show(); // Change d'onglet
    isModified = false; // Réinitialise le flag
    nextTabLink = null;
  }
  confirmModal.hide();
});
</script>

```



Résultat final :

- Tu as un système **UX-friendly** avec un **message entièrement personnalisable**.
 - Tu profites de **Bootstrap** pour le style et la logique d'onglet.
 - Tu évites l'alerte système basique.
-



Bonus (facultatif) :

Tu peux ajouter une animation ou un indicateur comme "Modifications non enregistrées" dans le titre de l'onglet actif si `isModified === true`, pour encore plus de clarté.

Souhaites-tu que je t'aide à intégrer ça dans un projet existant ou avec un framework spécifique (React, Vue, etc.) ?

You said:

Et dans une page vuejs 2 ?

ChatGPT said:

Très bien ! Voici comment faire la **même chose dans une page Vue.js 2**, avec des **onglets Bootstrap**, un **modal personnalisé** et la **gestion des modifications non enregistrées**.

Objectif :

- Tu utilises Vue.js 2 (avec le CDN ou dans un projet Vue CLI).
 - Tu as des onglets (Bootstrap ou customs).
 - Tu veux **bloquer le changement d'onglet** si un formulaire a été modifié.
 - Tu veux afficher un **modal personnalisé (Bootstrap)** pour demander confirmation.
-

Prérequis :

Assure-toi d'avoir :

- Vue.js 2 chargé
 - Bootstrap 5 (ou Bootstrap 4)
 - Bootstrap JS (nécessaire pour les modals)
-

Exemple Vue.js 2 avec modal de confirmation

1. HTML de base (dans `index.html` ou composant Vue)

```
<div id="app" class="container mt-5">
  <!-- Onglets -->
  <ul class="nav nav-tabs">
    <li class="nav-item" v-for="tab in tabs" :key="tab.id">
      <a
        class="nav-link"
        :class="{ active: activeTab === tab.id }"
        href="#"
        @click.prevent="onTabClick(tab.id)"
      >
        {{ tab.name }}
      </a>
    </li>
  </ul>
</div>
```

```

</ul>

<!-- Contenu des onglets -->
<div class="tab-content mt-3">
  <div v-show="activeTab === 'tab1'">
    <form @input="formModified = true">
      <input type="text" class="form-control" placeholder="Nom">
    </form>
  </div>
  <div v-show="activeTab === 'tab2'">
    <p>Contenu de l'onglet 2</p>
  </div>
</div>

<!-- Modal Bootstrap personnalisé -->
<div class="modal fade" id="confirmModal" tabindex="-1" role="dialog" aria-
hidden="true" ref="confirmModal">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Modifications non enregistrées</h5>
        <button type="button" class="close" data-bs-dismiss="modal" aria-
label="Fermer">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <p>Vous avez des modifications non enregistrées. Voulez-vous vraiment
changer d'onglet ?</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Annuler</button>
        <button type="button" class="btn btn-primary"
@click="confirmTabChange">Oui, continuer</button>
      </div>
    </div>
  </div>
</div>
</div>

```

2. JavaScript (Vue.js 2)

```

<script>
  new Vue({
    el: '#app',
    data: {
      tabs: [
        { id: 'tab1', name: 'Onglet 1' },
        { id: 'tab2', name: 'Onglet 2' },
      ],
      activeTab: 'tab1',
      formModified: false,
      nextTab: null,
      modalInstance: null,
    },
    mounted() {

```

```

    // Initialiser le modal Bootstrap
    this.modalInstance = new bootstrap.Modal(this.$refs.confirmModal);
  },
  methods: {
    onTabClick(tabId) {
      if (this.formModified) {
        this.nextTab = tabId;
        this.modalInstance.show();
      } else {
        this.activeTab = tabId;
      }
    },
    confirmTabChange() {
      this.activeTab = this.nextTab;
      this.formModified = false;
      this.nextTab = null;
      this.modalInstance.hide();
    }
  }
});
</script>

```



Résultat :

- L'utilisateur modifie le formulaire dans **Onglet 1**.
 - Il clique sur **Onglet 2** → un **modal Bootstrap** s'affiche.
 - Il peut confirmer ou annuler le changement d'onglet.
 - Le message est **entièrement personnalisable** (pas de beforeunload ici).
-



Astuces :

- Tu peux aussi utiliser `@change` ou `@input` plus finement pour chaque champ si besoin.
 - Si tu veux **remettre le formulaire à zéro** après changement d'onglet, tu peux ajouter `this.$refs.form1.reset()` ou équivalent.
 - Pour plus de modularité, tu peux splitter chaque tab dans un composant Vue.
-

Souhaites-tu que je t'aide à faire cette version dans un projet Vue CLI (avec `.vue` components) ?

You said:

oui

ChatGPT said:

Parfait, on va maintenant faire la **version Vue CLI** avec **composants .vue**, utilisant :

- **Vue.js 2**
 - **Bootstrap 5**
 - **Modal Bootstrap** personnalisé
 - **Gestion d'un formulaire modifié dans un onglet**
 - **Interception du changement d'onglet avec confirmation**
-



Architecture Vue CLI (simplifiée)

```
src/  
├── components/  
│   ├── Tab1.vue  
│   └── Tab2.vue  
├── App.vue  
└── main.js
```



Étape 1 : Installer Bootstrap

Si ce n'est pas encore fait :



Installer Bootstrap 5

```
npm install bootstrap
```

Puis dans `main.js` :

```
import Vue from 'vue';  
import App from './App.vue';  
  
import 'bootstrap/dist/css/bootstrap.min.css';  
import 'bootstrap/dist/js/bootstrap.bundle.min.js'; // important pour modals  
  
Vue.config.productionTip = false;  
  
new Vue({  
  render: h => h(App),  
}).$mount('#app');
```



Étape 2 : Composants d'onglets



Tab1.vue

```
<template>  
  <div>  
    <form @input="onInput" ref="form">
```

```

        <input type="text" class="form-control" placeholder="Nom">
    </form>
</div>
</template>

<script>
export default {
  name: 'Tab1',
  props: ['onModify'],
  methods: {
    onInput() {
      this.onModify(true); // Signale que le formulaire est modifié
    },
  },
}
</script>

```

Tab2.vue

```

<template>
  <div>
    <p>Contenu de l'onglet 2</p>
  </div>
</template>

<script>
export default {
  name: 'Tab2'
}
</script>

```

Étape 3 : Composant principal

App.vue

```

<template>
  <div class="container mt-5">
    <!-- Onglets -->
    <ul class="nav nav-tabs">
      <li class="nav-item" v-for="tab in tabs" :key="tab.id">
        <a
          href="#"
          class="nav-link"
          :class="{ active: activeTab === tab.id }"
          @click.prevent="handleTabClick(tab.id)"
        >
          {{ tab.name }}
        </a>
      </li>
    </ul>

    <!-- Contenu -->

```

```
<div class="tab-content mt-3">
  <Tab1 v-if="activeTab === 'tab1'" :onModify="setFormModified" />
  <Tab2 v-if="activeTab === 'tab2'" />
</div>

<!-- Modal Bootstrap -->
<div class="modal fade" id="confirmModal" tabindex="-1" ref="confirmModal">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Modifications non enregistrées</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        Vous avez des modifications non enregistrées. Voulez-vous changer d'onglet ?
      </div>
      <div class="modal-footer">
        <button class="btn btn-secondary" data-bs-dismiss="modal">Annuler</button>
        <button class="btn btn-primary" @click="confirmTabChange">Oui, continuer</button>
      </div>
    </div>
  </div>
</div>
</div>
</template>

<script>
import Tab1 from './components/Tab1.vue';
import Tab2 from './components/Tab2.vue';

export default {
  name: 'App',
  components: {
    Tab1,
    Tab2
  },
  data() {
    return {
      tabs: [
        { id: 'tab1', name: 'Onglet 1' },
        { id: 'tab2', name: 'Onglet 2' }
      ],
      activeTab: 'tab1',
      formModified: false,
      nextTab: null,
      modalInstance: null
    };
  },
  mounted() {
    // Bootstrap modal instance
    this.modalInstance = new bootstrap.Modal(this.$refs.confirmModal);
  },
  methods: {
    setFormModified(modified) {
      this.formModified = modified;
    }
  }
}
```

```
    },
    handleTabClick(tabId) {
      if (this.formModified) {
        this.nextTab = tabId;
        this.modalInstance.show();
      } else {
        this.activeTab = tabId;
      }
    },
    confirmTabChange() {
      this.activeTab = this.nextTab;
      this.formModified = false;
      this.nextTab = null;
      this.modalInstance.hide();
    }
  }
}
</script>
```



Résultat :

- Tu as deux onglets avec composants Vue.
 - Si tu modifies le formulaire dans `Tab1.vue` et veux aller vers `Tab2.vue`, un **modal personnalisé** apparaît.
 - Si tu confirmes, le changement d'onglet s'effectue.
-



Bonus idées :

- Tu peux gérer plusieurs onglets avec des états séparés (`formModified.tab1`, etc.).
 - Tu peux faire un mixin ou une store Vuex pour gérer le suivi de modifications dans toute l'app.
 - Tu peux utiliser `v-show` au lieu de `v-if` pour garder l'état des composants montés.
-

Souhaites-tu que je t'envoie un projet zip prêt à tester, ou tu veux ajouter d'autres fonctionnalités (comme réinitialiser le formulaire après changement, ou avertir aussi sur fermeture d'onglet navigateur) ?