Les bulles d'aide

Author: Benjamin POUSSIN
Contact: poussin@codelutin.com
Revision: 1.3
Date: 2003-05-09

Sommaire

Dans ce document nous tâcherons de recenser les besoins essentiels pour une fonctionnalité de bulle d'aide. Puis nous verrons de quelle manière il faut envisager leur intégration dans Glasnost, ainsi que leur apparence.

1 Différents besoins

Même si le nom **bulle d'aide** devrait représenter des bulles qui affichent de l'aide, la librairie développée devra pouvoir servir pour d'autres tâches que l'affichage des bulles d'aides.

1.1 Fonctionnalités attendues

- affichage d'une aide contextuelle lorsque l'on positionne la souris sur un élément de la page.
- affichage d'une aide lorsque qu'un élément de la page prend le focus, par exemple lorsque le passage d'un champ à un autre dans un formulaire se fait par les touches du clavier et non a la souris.
- affichage d'une description plus complète dans le calendrier lorsque la souris se positionne sur le titre de l'événement.
- affichage d'une description plus complète d'une tâche lorsque la souris se positionne sur la tâche dans la liste des tâches.
- affichage d'une description plus complète lorsque l'on survole un projet avec la souris.
- affichage complet d'une note, d'une tâche ou d'un projet lorsque l'on positionne la souris sur le lien correspondant.
- affichage complet d'une alerte d'une tâche ou d'un projet lorsque l'on position la souris sur le lien de l'alerte.

On peut donc simplifier le besoin en: affichage d'éléments *HTML* dans une boîte qui apparaît en surimpression du contenu de la page.

1.2 Quelques questions

- comment déterminer l'élément qui active la bulle ?
- où doit être positionnée la bulle par rapport à l'élément qui l'active ?
- quels sont les conditions qui font que la bulle disparaît.

1.3 Quelques contraintes supplémentaires

- le code *HTML* généré doit être valide et bien formé.
- il ne doit pas apparaître de code *ECMAScript* [1] dans la page *HTML*
- l'utilisation de l'*ECMAScript* doit se limiter à l'inclusion d'un lien vers un fichier contenant le code *ECMAScript*.
- la page *HTML* doit apparaître convenablement même si le fichier *ECMAScript* n'est pas chargé.

2 Implémentation

En regard des différentes utilisations qui seront faites des bulles, il est souhaitable de trouver une façon de faire qui permettent d'utiliser la même librairie pour tous les cas d'utilisation, et de ne pas avoir de code spécifique pour un cas d'utilisation particulier.

L'avantage de cette approche est de ne pas demander la réécriture du code *ECMAScript* à chaque fois que l'on veut réutiliser les bulles pour une nouvelle fonctionnalité.

La contrainte qui indique de ne pas mettre de code *ECMAScript* dans la page HTML généré à part l'inclusion d'un fichier implique que le code *ECMAScript* doit lui même ajouter le code dont il a besoin dans la page. Ceci se fait en plusieurs étapes car la page doit être complètement chargée avant que la librairie de bulle ne rajoute son code.

Les différentes étapes:

- ajout de la ligne d'inclusion dans l'entête du fichier HTML généré.
- le code de ce fichier *ECMAScript* est directement interprété, il faut donc attendre que la page soit complètement chargée dans le navigateur pour que l'ajout de nouveaux éléments se fasse. Ceci est possible en ajoutant un événement **load** sur la fenêtre.
- lorsque la fenêtre est chargée recherche des bulles et des éléments qui doivent les actionner et ajout des éléments manquants.

Après ces différentes étapes la page HTML est complètement fonctionnelle. Les éléments à ajouter sont de plusieurs types. Nous avons du code de feuilles de style (*CSS* [2]), du code *ECMAScript*, et enfin des éléments *HTML*

Dans le code HTML une bulle d'aide doit être du code *HTML* valide et bien formé, donc le plus simple pour marquer qu'un élément *HTML* est en fait une bulle est d'utiliser les **class** *HTML*. Cette approche présente aussi l'avantage de pouvoir désactiver les bulles sans aucun code *ECMAScript*, en utilisant juste des *CSS*.

Exemple de code *HTML* utilisable pour les bulles:

```
<a href="lien.html">Un lien
  <div class="tooltip">
      Ceci est une petite aide pour
      expliquer ce qui se passe lors
      de l'appuie sur le lien.
  </div>
</a>
```

[1] *ECMAScript* est la normalisation du *JavaScript* et du *JScript*.

[2] CSS Cascading Style Sheet

Exemple de code *CSS* pour désactiver les bulles:

```
.tooltip {
  style.visibility = 'hidden';
  style.display = 'none';
}
```

2.1 Relation entre la bulle et l'élément qui doit l'activer

Nous avons maintenant une façon de déclarer les bulles dans le code *HTML* et la possibilité de les cacher sans utiliser d'*ECMAScript*. Il nous faut définir une façon de lier une bulle avant l'élément qui l'active. Il faut que ce système conserve la validité du code *HTML* et soit le plus simple possible.

Dans l'exemple précédent nous voyons que l'élément *HTML* qui doit activer la bulle est en fait l'élément père de la bulle. Nous utiliserons donc cette caractéristique comme valeur par défaut pour retrouver l'élément d'activation d'une bulle en prenant le père de la bulle par défaut.

Ceci est le comportement souhaité dans la majorité des cas. Mais il existe certains cas où le père ne doit pas être l'élément d'activation de la bulle.

Par exemple:

```
<span class="field">
  <span class="label">Le titre</span>
  <div class="field-description tooltip">
    Ceci est la description du champ title
    qui sert en même temps de bulle d'aide.
  </div>
  <input type="text" name="title"/>
</span>
```

Dans cet exemple si nous utilisons la notion de père pour activer la bulle d'aide, celle-ci s'activera alors sur tous les éléments et non pas seulement sur le champ **input** sur lequel il est logique que la bulle s'active. Car elle indique une aide pour remplir le champ. Si nous souhaitons utiliser le comportement d'activation par le père il faudrait alors mettre la bulle dans l'élément **input** ce qui n'est pas la sémantique souhaitée. Il faut donc imaginer un autre moyen pour choisir l'élément d'activation.

Il faut donc pouvoir créer un lien entre la bulle et l'élément qui l'activera. Ce lien doit toujours être du *HTML* valide.

Une première idée serait d'ajouter un attribut sur la bulle qui indiquerait l'identifiant d'un autre élément qui serait l'élément qui activerait la bulle.

Par exemple:

```
<span class="field">
  <span class="label">Le titre</span>
  <div class="field-description tooltip" activate="title">
    Ceci est la description du champ title
    qui sert en même temps de bulle d'aide.
  </div>
  <input id="title" type="text" name="title"/>
</span>
```

Le problème est que ceci n'est pas du code *HTML* valide car l'attribut **activate** n'existe pas dans la spécification. Il faut donc utiliser un attribut existant. Pour cela nous pouvons utiliser l'attribut **id** mais comme deux éléments ne doivent pas avoir le même identifiant nous préfixerons l'identifiant de la bulle par **for-**

Par exemple:

```
<span class="field">
  <span class="label">Le titre</span>
  <div class="field-description tooltip" id="for-title">
    Ceci est la description du champ title
    qui sert en même temps de bulle d'aide.
```

```
    </div>
    <input id="title" type="text" name="title"/>
  </span>
```

Grâce à ce mécanisme, nous pouvons rechercher toutes les bulles grâce à leur *class* particulière **tooltip**. Ensuite pour chaque bulle regarder si elle contient un attribut **id** commençant par **for-** et qui correspond à l'identifiant d'un autre élément de la page. Si cet autre élément est trouvé alors il est utilisé pour activer cette bulle, sinon on prend le comportement par défaut qui est d'activer la bulle sur le père.

Nous avons donc maintenant deux moyens pour choisir l'élément d'activation de la bulle. Le premier qui est simple et recouvre la majorité des cas, et le second qui est un peu plus complexe et qui permet de couvrir les cas restants.

2.2 Positionnement des bulles d'aides

Il nous reste à définir la position des bulles d'aide. La bulle ne doit pas recouvrir l'élément pour lequel elle indique une aide. Le mieux et le plus naturel est de la faire apparaître sous l'élément qui l'active.

Il faudra lors de l'implantation faire attention à ce que cette bulle reste le plus possible dans la zone d'affichage du navigateur, c'est à dire que la personne n'ait pas à utiliser les barres de défilement pour visualiser la bulle. Elle devra donc calculer la taille qu'elle devrait faire normalement, calculer la place qu'elle a pour s'afficher, et si possible modifier sa taille pour s'afficher entièrement dans l'écran.