

Model objet

Les objets Javascript ne supportant pas l'observation des changements de leurs propriétés, une application Ember utilisera des objets [Ember.Object](#) pour mettre en place le binding des propriétés en lieu et place des objets javascript standards.

De la même façon, Ember étend l'objet Array javascript par la classe [Ember.Enumerable](#).

Création de classes

La création de classe se fait en utilisant la méthode `extend` sur la classe `Ember.Object` :

```
import EmberObject from '@ember/object';

const Person = EmberObject.extend({
  say(thing) {
    alert(thing);
  }
});
```

Instanciation

L'instanciation d'un objet se fait par appel de la méthode **create** sur la classe.

```
let person = Person.create();
person.say('Hello'); // alerts " says: Hello"
```

La méthode `create` accepte en paramètre un objet js permettant d'initialiser/créer des membres :

```
import EmberObject from '@ember/object';

const Person = EmberObject.extend({
  helloWorld() {
    alert(`Hi, my name is ${this.get('name')}`);
  }
});

let tom = Person.create({
  name: 'Tom Dale'
});

tom.helloWorld(); // alerts "Hi, my name is Tom Dale"
```

Héritage/surdéfinition

- L'héritage est mis en place grâce à la méthode **extend()**
- La surdéfinition par l'appel de **_super()**

```
import EmberObject from '@ember/object';

const Person = EmberObject.extend({
  say(thing) {
    alert(`${this.get('name')} says: ${thing}`);
  }
});

const Soldier = Person.extend({
  say(thing) {
    // this will call the method in the parent class (Person#say), appending
    // the string ', sir!' to the variable `thing` passed in
    this._super(`${thing}, sir!`);
  }
});

let yehuda = Soldier.create({
  name: 'Yehuda Katz'
});

yehuda.say('Yes'); // alerts "Yehuda Katz says: Yes, sir!"
```

Initialisation/construction

Quand un objet est instancié, la méthode **init()** de sa classe est invoquée automatiquement.

```
import EmberObject from '@ember/object';

const Person = EmberObject.extend({
  init() {
    alert(`${this.get('name')}, reporting for duty!`);
  }
});

Person.create({
  name: 'Stefan Penner'
});

// alerts "Stefan Penner, reporting for duty!"
```

La méthode `init` permet d'initialiser chaque instance.

getters/setters

L'accès aux propriétés d'un objet doit se faire en passant par les accesseurs **get()** et **set()**, faute de quoi computed properties et observers ne sont pas sollicités et ne mettent pas à jour les données (dans un template par exemple)

```
import EmberObject from '@ember/object';

const Person = EmberObject.extend({
  name: 'Robert Jackson'
});

let person = Person.create();

person.get('name'); // 'Robert Jackson'
person.set('name', 'Tobias Fünke');
person.get('name'); // 'Tobias Fünke'
```

From:

<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:

<http://slamwiki2.kobject.net/richclient/emberjs/modelobjet>

Last update: **2019/08/31 14:21**

