Prototype

# Prototipo

Todo objeto está vinculado a un objeto prototipo del que puede heredar propiedades. Todos los objetos creados a partir de literales de objeto están vinculados a Object.prototype, un objeto que viene de serie con JavaScript.

Cuando creas un nuevo objeto, puedes seleccionar el objeto que debe ser su prototipo. El mecanismo que proporciona JavaScript para hacer esto es desordenado y complejo, pero puede simplificarse significativamente. Añadiremos un método create a la función Object. El método beget crea un nuevo objeto que utiliza un objeto antiguo como prototipo. En el próximo capítulo hablaremos mucho más sobre las funciones.

if (typeof Object.create !== 'function') {  
 Object.create = function (o) {  
 var F = function () {};  
 F.prototype = o;  
 return new F();  
 };  
}  
var another\_stooge = Object.create(stooge);

El enlace prototipo no tiene ningún efecto sobre la actualización. Cuando hacemos cambios en un objeto, el prototipo del objeto no se toca:

another\_stooge['first-name'] = 'Harry';  
another\_stooge['middle-name'] = 'Moses';  
another\_stooge.nickname = 'Moe';

El enlace prototipo sólo se utiliza en la recuperación. Si intentamos recuperar el valor de una propiedad de un objeto, y si el objeto carece del nombre de la propiedad, entonces JavaScript intenta recuperar el valor de la propiedad del objeto prototipo. Y si ese objeto carece de la propiedad, entonces pasa a *its* prototipo, y así sucesivamente hasta que el proceso llega finalmente a Object.prototype. Si la propiedad deseada no existe en ninguna parte de la cadena de prototipos, entonces el resultado es el valor undefined. A esto se le llama *delegation*.

La relación prototipo es una relación dinámica. Si añadimos una nueva propiedad a un prototipo, esa propiedad será visible inmediatamente en todos los objetos que se basen en ese prototipo:

stooge.profession = 'actor';  
another\_stooge.profession // 'actor'

Veremos más sobre la cadena de prototipos en [el Capítulo 6](ch06.html).