Javascript difference between uber and prototype

Asked 9 years, 11 months ago Modified 9 years, 11 months ago

Viewed 3k times

5

I'm relatively new in javaScript and I was doing some inheritance. I thought I know what prototype is but then i met with uber method. Now I don't know the difference between those two. I know that uber is like super in java and that's all. But then prototype is something that bothers me. If you can give me some simple example of using those two I would appreciate it very.

javascript

Share

Improve this question

Follow

asked Jan 14, 2015 at 11:07

anicicn

193 ● 5 ● 15

2 As far as I am aware, there is no `uber` method in JavaScript unless you're using a library which adds this method to objects. Are you using a library by any chance? – Qantas 94 Heavy Jan 14, 2015 at 11:17

Well I'm using one. It might be defined in it. But still I don't know the difference assuimng I know what uber is. It's just a

pointer to a parent's object – anicicn Jan 14, 2015 at 11:22

1 I've added an answer about the `uber` method Douglas Crockford uses in his inheritance tutorials assuming that you meant this. It seems to be the most popular example. Where exactly did you discover `uber`? – Alex Wolf Jan 14, 2015 at 11:23

I'm not sure Crockford is the best way to learn prototype as I've never seen him implement "classical inheritance" correctly. The following answer covers constructor functions and prototype: stackoverflow.com/a/16063711/1641941 hope it helps – HMR Jan 15, 2015 at 1:26

1 Answer

Sorted by: Highest score (default)

12

`uber` is just a *sugar* method Douglas Crockford created in his examples of inheritance in JavaScript which should help the devoloper when working with the very, very flexible nature of JavaScripts prototypal inheritance.

This method doesn't exist in native JavaScript.

He explains the *sugar* methods he uses in detail here.

In his examples he defines the `uber` method as a helper method to access the parent implementation of a method.

Let's assume that you have a "class" (I use this term to ease the example; strictly spoken there are no classes in JavaScript) `Human` which has a `walk` method. If you now "extend" this class in an `Infant` class, you could

overwrite `walk` in such a way that the infant only crawls since it can't walk.

It's obviously not a great example but I hope you get the point.

In such a case you could use Douglas Crockfords `uber` method to access the `Human` implementation of `walk` in the `Infant` "class".

To compare JavaScripts native `prototype` object and Douglas Crockfords `uber` method would make no sense, since both serve completly different purposes.

If you want more information on JavaScripts `prototype` you can take a look at this question.

Share Improve this answer

Follow

edited May 23, 2017 at 12:00

Community Bot

1 ● 1

answered Jan 14, 2015 at 11:20

Alex Wolf

20.1k ● 4 ● 52 ● 73