RESPONSIVE WEB DESIGN

Quick Tip: Don't Forget the Viewport Meta Tag

by Ian Yates *26 Jun 2013* *109 Comments* English ▼

120 113

This post is part of a series called Strange and Unusual HTML.

Explaining the "Details" and "Summary" Elements

I remember my maiden voyage into responsive web design; I'd used a classic grid, wrestled with a flexible layout, and tackled media queries for the first time. Stretching and shrinking the browser window resulted in the satisfying sight of my design responding to its surroundings. Then I tested it on a mobile. It didn't work - I was looking at a shrunken down version of the full-screen design. The solution, as it turned out, was simple..

Note: This tutorial was first published in February 2012, but it's often used as a reference in other tutorials (and things have changed) so I felt it warranted an update.

The Crux

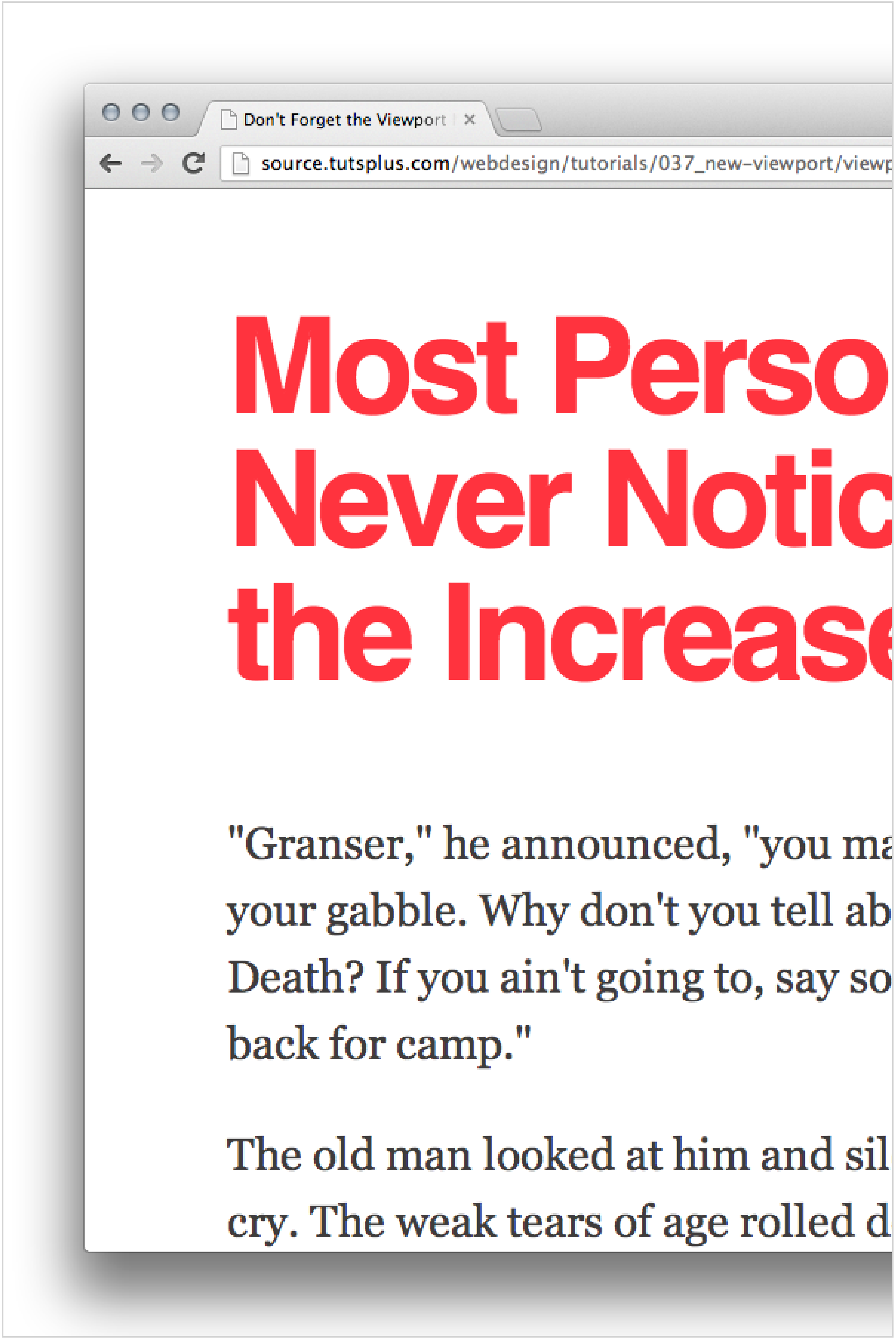
If you read nothing else within this post, take one bit of advice away: **if you're designing flexibly, use the viewport meta tag in your <head>**. In its basic form, it will set you up for cross-device layout peace of mind:

```html
<meta name="viewport" content="initial-scale=1">
```

The Problem

Let's use an example layout I've whipped together. Take a look; you'll see it shrink and grow as you alter the size of your browser. There's also a single media query which makes the text larger and gives the heading a bright `#ff333e` color on wider screens. Lovely.

Here's how it looks in OSX Chrome:

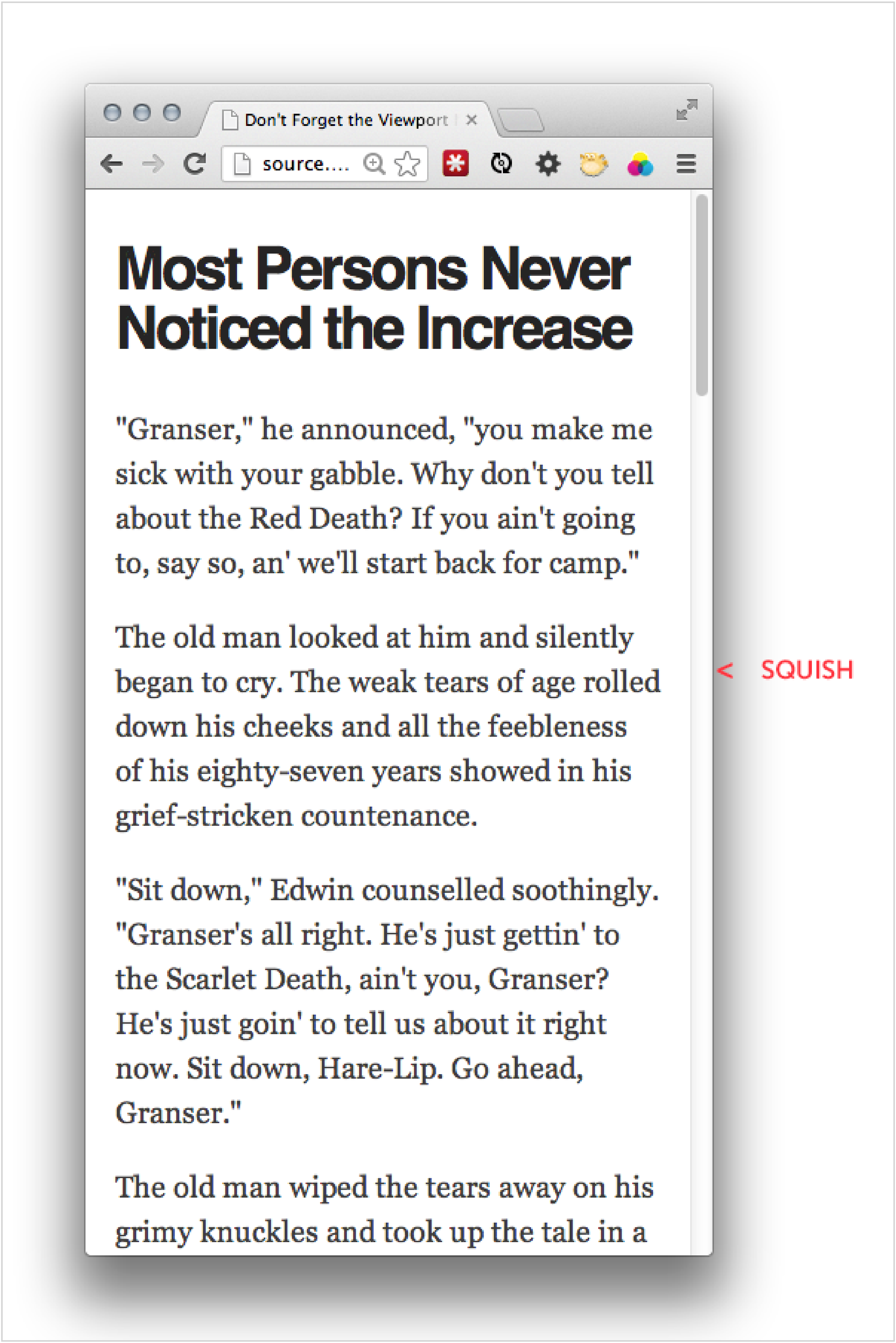

Don't Forget the Viewport
source.tutsplus.com/webdesign/tutorials/037_new-viewport/viewp

Most Perso
Never Notic
the Increase

"Granser," he announced, "you ma
your gabble. Why don't you tell ab
Death? If you ain't going to, say so
back for camp."

The old man looked at him and sil
cry. The weak tears of age rolled d

And here's how it appears when the browser window is squished:



Most Persons Never Noticed the Increase

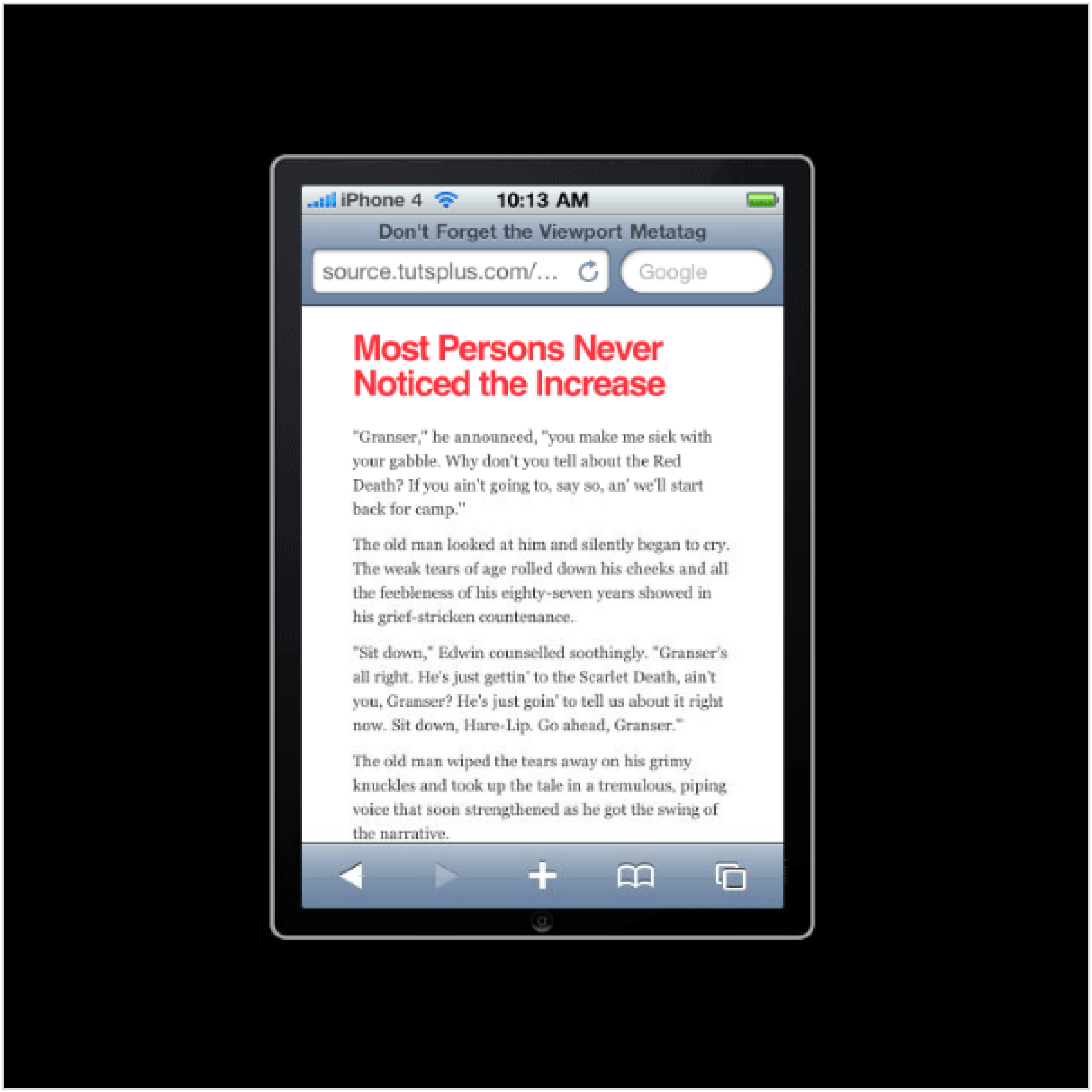
"Granser," he announced, "you make me sick with your gabble. Why don't you tell about the Red Death? If you ain't going to, say so, an' we'll start back for camp."

The old man looked at him and silently began to cry. The weak tears of age rolled down his cheeks and all the feebleness of his eighty-seven years showed in his grief-stricken countenance.

"Sit down," Edwin counselled soothingly. "Granser's all right. He's just gettin' to the Scarlet Death, ain't you, Granser? He's just goin' to tell us about it right now. Sit down, Hare-Lip. Go ahead, Granser."

The old man wiped the tears away on his grimy knuckles and took up the tale in a

Now let's see how that looks on a smartphone (in this case iOS Safari on an iPhone 4):



Most Persons Never Noticed the Increase

"Granser," he announced, "you make me sick with your gabble. Why don't you tell about the Red Death? If you ain't going to, say so, an' we'll start back for camp."

The old man looked at him and silently began to cry. The weak tears of age rolled down his cheeks and all the feebleness of his eighty-seven years showed in his grief-stricken countenance.

"Sit down," Edwin counselled soothingly. "Granser's all right. He's just gettin' to the Scarlet Death, ain't you, Granser? He's just goin' to tell us about it right now. Sit down, Hare-Lip. Go ahead, Granser."

The old man wiped the tears away on his grimy knuckles and took up the tale in a tremulous, piping voice that soon strengthened as he got the swing of the narrative.

The first thing you'll notice is the red heading; a dead giveaway that we're not looking at the narrow layout that we would expect. We're actually looking at a zoomed-out version.

iOS Safari assumes a webpage is 980px wide, zooming out in order to fit the whole lot within the available (iPhone 4) 320px. The content is then much less readable, unless you zoom in.

Why?

As they say, assumption is the mother of all… something but that's exactly what mobile browsers have to do if you don't specifically instruct them. When you visit a website via a mobile browser it will assume that you're viewing a big desktop experience and that you want to see all of it, not just the top left corner. It will therefore set the viewport width at (in the case of iOS Safari) 980px, shoe-horning everything into its little display.

The Viewport Meta Tag

Enter the viewport meta tag, introduced by Apple, then adopted and developed further by others.

It looks like this:

```
<meta name="viewport" content="">
```

Within the `content=""` you can enter a load of comma delimited values, but we're going to to focus on the fundamental ones for now.

For example, if your mobile design is purposely laid out at 320px you can specify the viewport width:

```
<meta name="viewport" content="width=320">
```

For flexible layouts it's more practical to base your viewport width on the device in question, so to match your layout width to the device width you'd enter:

```
<meta name="viewport" content="width=device-width">
```

To be extra certain that your layout will be displayed as you intended it you can also set the zoom level. This, for example:

```
<meta name="viewport" content="initial-scale=1">
```

..will ensure that upon opening, your layout will be displayed properly at 1:1 scale. No zooming will be applied. You could even go further and prevent any zooming by the user:

```
<meta name="viewport" content="maximum-scale=1">
```

Note: Before applying the maximum-scale parameter, consider whether you should really be preventing your users from zooming in. Can they read everything as well as possible?

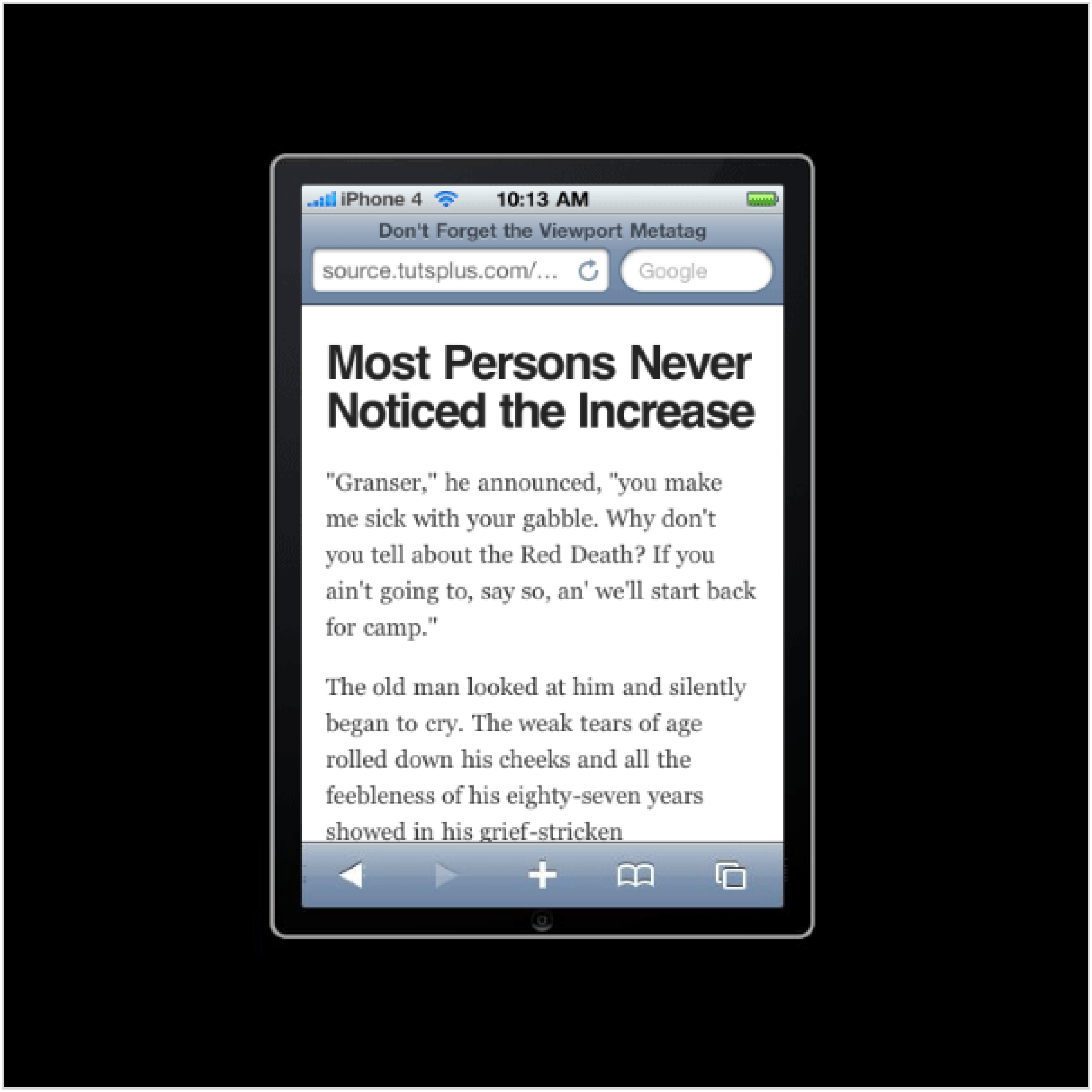
> *Using the viewport meta tag the wrong way can hinder users with visual problems accessing your website*
>
> *- The Accessibility Project*

Put it All Together

These values together give us a great default to work with:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Let's see how it affects our example



Most Persons Never Noticed the Increase

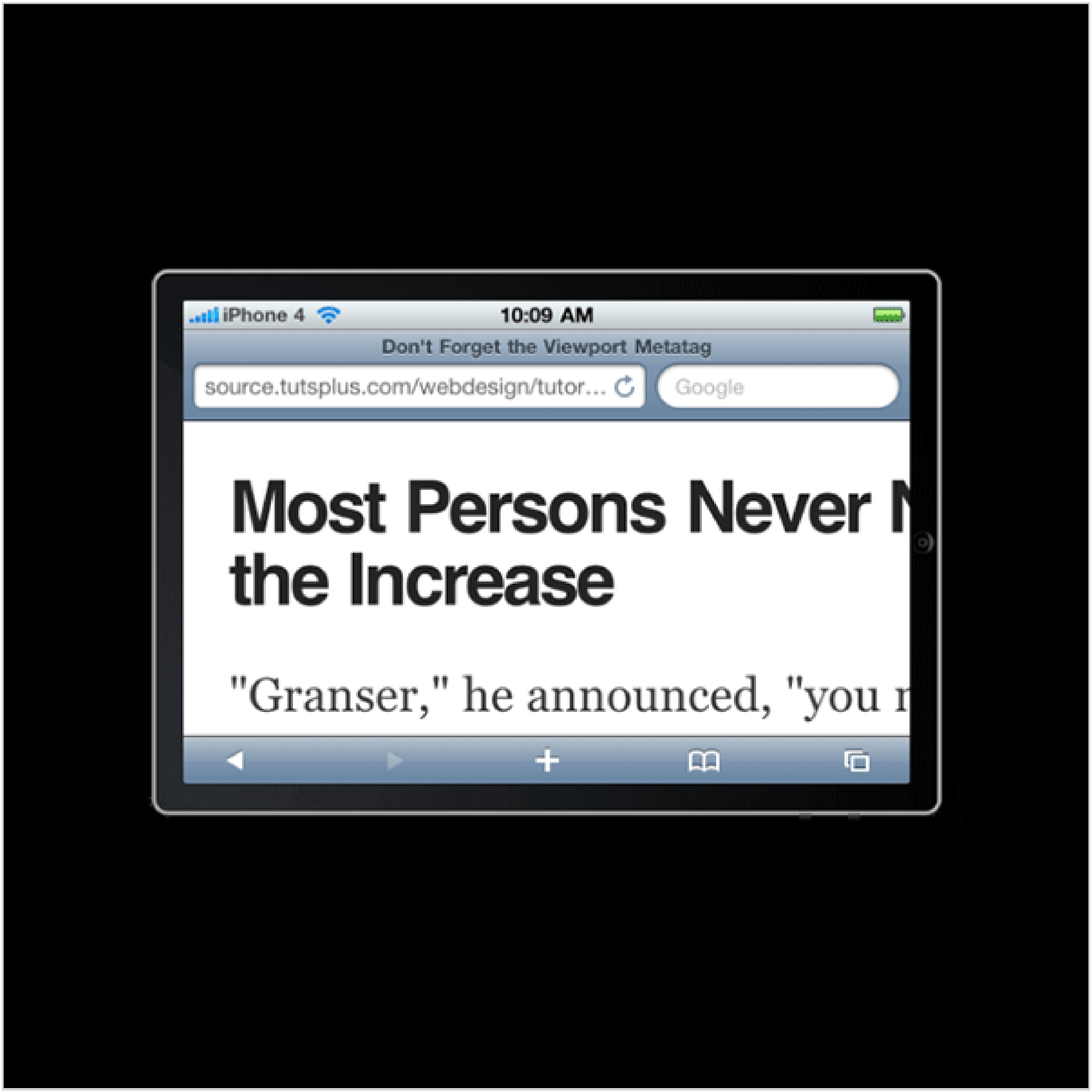
"Granser," he announced, "you make me sick with your gabble. Why don't you tell about the Red Death? If you ain't going to, say so, an' we'll start back for camp."

The old man looked at him and silently began to cry. The weak tears of age rolled down his cheeks and all the feebleness of his eighty-seven years showed in his grief-stricken

As you can see, everything remains at the correct scale. The text is wrapping instead of shrinking, and our visual helper (the red heading) has disappeared.

But!

Hat tip to Jason for pointing this out in the comments. By stating that the width of your layout is equal to the device width, you'll run into problems when a device is used in landscape. Take a look at how an iPhone 4 handles our demo, in landscape:


iPhone 4
10:09 AM
Don't Forget the Viewport Metatag
source.tutsplus.com/webdesign/tutor...
Google

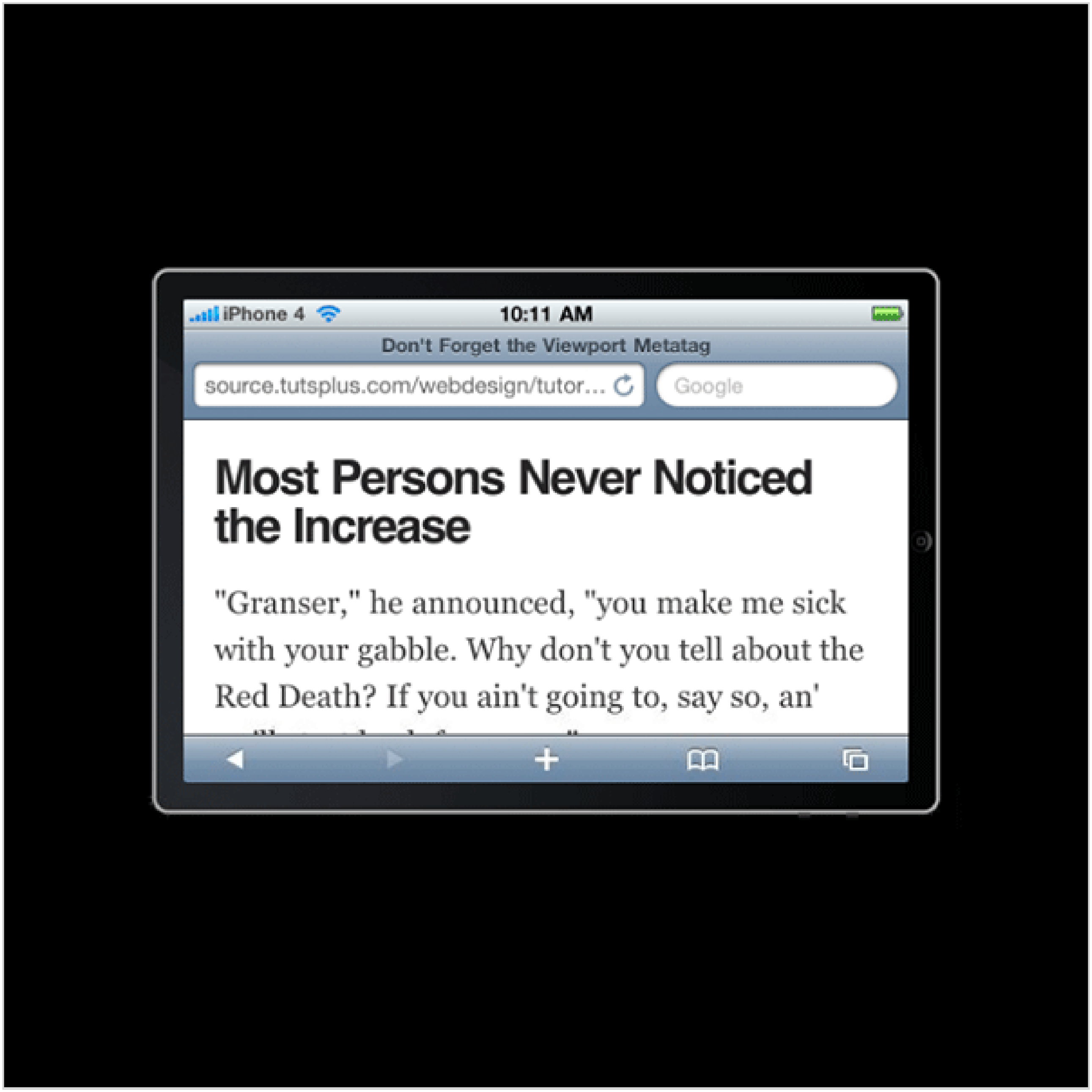
Most Persons Never N
the Increase

"Granser," he announced, "you r

Ruined.

The browser feels obliged to zoom in, in order to match the width of the page.

To get around this, we could, as Jason points out, use JavaScript to conditionally choose which metatag attributes to go with, but the simplest solution seems to be simply to ignore the `width` altogether. By only setting the `initial-scale` the width is inferred. This is how our new, simplified demo appears in landscape:


iPhone 4
10:11 AM
Don't Forget the Viewport Metatag
source.tutsplus.com/webdesign/tutor...
Google

Most Persons Never Noticed the Increase

"Granser," he announced, "you make me sick with your gabble. Why don't you tell about the Red Death? If you ain't going to, say so, an'

Much better.

Not the Standard You're Looking For

The viewport metatag was Apple's solution to the problem. It was adopted quickly by other platforms, but it was never put forward by the W3C. Microsoft brought this to light when they chose for IE10 to ignore the viewport metatag under certain circumstances. Instead, they opted to use CSS Device Adaptation, which *is* what the W3C are leaning on.

In short, similar viewport properties are defined within CSS using the @viewport rule, instead of within the HTML.

```
@viewport{
    zoom: 1.0;
    width: device-width;
}
```

Or, in accordance to our newer *don't specify device width* approach, we could set the following:

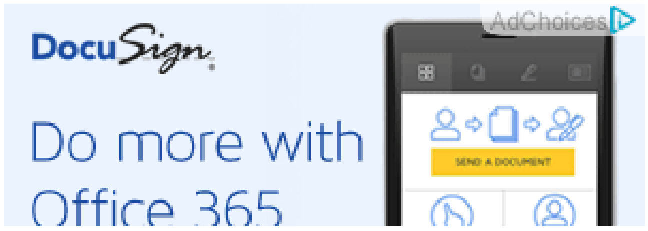
```
@viewport{
    zoom: 1.0;
    width: extend-to-zoom;
}
```

Given that it's a work in progress, IE10 requires the prefixed version of the proposal, which looks something like this:

```
@-ms-viewport{
    width: extend-to-zoom;
    zoom: 1.0;
}
```

It's a far more elegant solution than the metatag, but is a long way from being fully supported. Slot it into your CSS now, to make sure your responsive design behaves itself in IE10 "snap mode", and keep an eye on its progress in the future. This is where viewport control is headed.

You can read more about this in Tim Kadlec's post IE10 Snap Mode and Responsive Design.

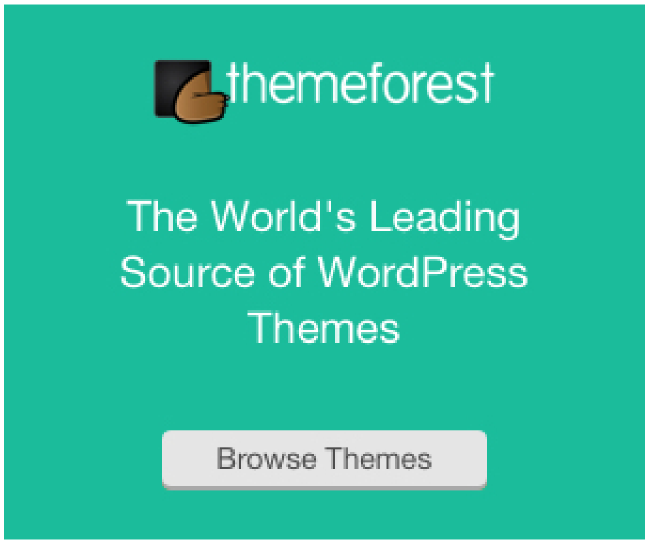


Conclusion

Responsive web design isn't just for people who enjoy watching their browsers grow and shrink, it's about catering for as many different devices, screens and resolutions as possible! Throw the viewport meta tag into your `<head>`, plus the @viewport rule into your CSS when you're building flexible layouts and you're good to go.

Further Reading

- Apple documentation: Using the Viewport Meta Tag
- Mozilla: Using the viewport meta tag to control layout on mobile browsers
- quirksmode.org: A tale of two viewports
- w3.org Mobile Web Application Best Practices: Use Meta Viewport Element To Identify Desired Screen Size
- Quick Tip: Never use maximum-scale=1.0 on the A11y Project
- Tim Kadlec explains IE10 Snap Mode and Responsive Design
- The @viewport rule proposal



Advertisement