HTML lists

From Web Education Community Group

Contents

Introduction

Lists are used to group related pieces of information together, so they are clearly associated with each other and easy to read. In modern web development lists are workhorse elements, frequently used for navigation as well as general content.

Lists are good from a structural point of view as they help create a well-structured, more accessible, easy-to-maintain document. They are also useful for a purely practical reason — they give you extra elements to attach CSS styles to, for a whole variety of styling (we'll get on to CSS later in the course - check out the Styling lists and links first, and then visit the [Web Standards Curriculum] table of contents for the full list of CSS articles available.).

In this part, we will cover the different list types available in HTML, when and how you should use them, with a couple of practical examples at the end.

The three list types

There are three list types in HTML:

- **unordered list** — used to group a set of related items, in no particular order.
- **ordered list** — used to group a set of related items, in a specific order.
- **description list** — used to display name/value pairs such as terms and their definitions, or times and

events.

Each one has a specific purpose and meaning — they are not interchangeable.

Unordered lists

Unordered lists, or bulleted lists, are used when a set of items can be placed in any order. An example is a shopping list:

- milk
- bread
- butter
- coffee beans

These items are all part of one list, however, you could put the items in any order and the list would still make sense:

- bread
- coffee beans
- milk
- butter

You can use CSS to change the bullet to one of several default styles, use your own image, or even display the list without bullets—we'll look at how to do that in the Styling lists and links article.

Unordered list markup

Unordered lists use one set of `<ul></ul>` tags, wrapped around many sets of `<li></li>`:

```
<code><ul>
  <li>bread</li>
  <li>coffee beans</li>
  <li>milk</li>
  <li>butter</li>
</ul></code>
```

Ordered lists

Ordered lists, or numbered lists, are used to display a list of items that need to be placed in a specific order. An example would be cooking instructions, which must be completed in order for the recipe to work:

1. Gather ingredients
2. Mix ingredients together
3. Place ingredients in a baking dish
4. Bake in oven for an hour
5. Remove from oven
6. Allow to stand for ten minutes
7. Serve

If the list items were moved around into a different order, the information would no longer make sense:

1. Gather ingredients
2. Bake in oven for an hour
3. Serve
4. Remove from oven

5. Place ingredients in a baking dish
6. Allow to stand for ten minutes
7. Mix ingredients together

Ordered lists can be displayed with one of several numbering or alphabetic systems—that is, letters or numbers. The default in most browsers is decimal numbers, but there are more options:

- Letters
  - Lowercase ascii letters (a, b, c…)
  - Uppercase ascii letters (A, B, C…).
  - Lowercase classical Greek: (έ, ή, ί…)
-
- Numbers
  - Decimal numbers (1, 2, 3…)
  - Decimal numbers with leading zeros (01, 02, 03…)
  - Lowercase Roman numerals (i, ii, iii…)
  - Uppercase Roman numerals (I, II, III…)
  - Traditional Georgian numbering (an, ban, gan…)
  - Traditional Armenian numbering (mek, yerku, yerek…)

Again, you can use CSS to change the style of your ordered lists.

Ordered list markup

Ordered lists use one set of `<ol></ol>` tags, wrapped around many sets of `<li></li>`:

```
<code><ol>
  <li>Gather ingredients</li>
  <li>Mix ingredients together</li>
  <li>Place ingredients in a baking dish</li>
  <li>Bake in oven for an hour</li>
  <li>Remove from oven</li>
  <li>Allow to stand for ten minutes</li>
  <li>Serve</li>
</ol></code>
```

Beginning ordered lists with numbers other than 1

It is possible to get an ordered list to start with a number other than 1 (or i, or I, etc.). This is done using the `<start>` attribute, which takes a numeric value, even if you're using CSS to change the the list counters to be alphabetic, roman or. This is useful if you have a single list of items, but you want to break the list up with some kind of note, or some other related information. For example, we could do this with the previous example:

```
<code><ol>
  <li>Gather ingredients</li>
  <li>Mix ingredients together</li>
  <li>Place ingredients in a baking dish</li>
</ol>

<p class="note">Before you place the ingredients in the baking dish, preheat the oven to 180 degrees centigrade,

<ol start="4">
  <li>Bake in oven for an hour</li>
  <li>Remove from oven</li>
  <li>Allow to stand for ten minutes</li>
  <li>Serve</li>
</ol></code>
```

This gives the following result:

1. Gather ingredients
2. Mix ingredients together
3. Place ingredients in a baking dish

Before you place the ingredients in the baking dish, preheat the oven to 180 degrees centigrade/350 degrees fahrenheit in readiness for the next step

4. Bake in oven for an hour
5. Remove from oven
6. Allow to stand for ten minutes
7. Serve

Note that this attribute was deprecated in HTML 4, so it will make your page not validate if you are using an HTML4 strict doctype. If you want to make use of such functionality in an HTML4 strict page, and it absolutely has to validate, you can do it using CSS Counters (http://dev.opera.com/articles/view/automatic-numbering-with-css-counters/) instead. `<start>` has however been reinstated in HTML5, which is a good thing, as it is useful.

Description lists

Description lists associate specific names and their values within a list, for example items in an ingredient list and their descriptions, article metadata such as authors and categories and their values, or competition winners and the years in which they won. Let's explore the ingredient list example a bit more deeply. You could write an ingredients list like so:

```
milk

A white, liquid dairy product.

bread

A baked food made of flour or meal.

butter

A yellow, solid dairy product.

coffee beans

The seeds of the fruit from certain coffee trees.
```

Note: In HTML4, description lists were called definition lists, so if you hear that term mentioned in conversation, then just assume you are talking about a description list. HTML5 has changed the definition of this type of list to the more general "description list" because designers and developers were all too often using description lists to mark up name/value groups that weren't items and descriptions.

You can have as many name-value groups as you like, but there must be at least one name and at least one value in each pair. You can associate more than one value with a single name, or vice versa. For example, the term "coffee" can have several meanings, and you could show them one after the other:

```
coffee

a beverage made from roasted, ground coffee beans
a cup of coffee
a social gathering at which coffee is consumed
a medium to dark brown colour
```

Alternatively you can have more than one name with the same value. This is useful to show variations of a

term, all of which have the same meaning:

```
soda
pop
fizzy drink
cola

a sweet, carbonated beverage.
```

Description lists are different from the other kinds of list, as they use names and values instead of list items. You wrap a description list in one set of `<dl></dl>` elements, wrapped around groups of `<dt></dt>` (name) and `<dd></dd>` (value) tags. You must pair at least one `<dt></dt>` with at least one `<dd></dd>`; a `<dt></dt>` should always come first in the source order.

A simple description list of single names with single values would look like this:

```
<dl>
  <dt>Name</dt>
  <dd>Value</dd>
  <dt>Name</dt>
  <dd>Value</dd>
  <dt>Name</dt>
  <dd>Value</dd>
</dl>
```

This is rendered as follows:

```
Name
  Value
Name
  Value
Name
  Value
```

In this example, we associate more than one value with a name, and vice versa:

```
<dl>
  <dt>Name</dt>
  <dd>Value that applies to the preceding name</dd>
  <dt>Name</dt>
  <dt>Name</dt>
  <dd>Value that applies to both of the preceding name</dd>
  <dt>Name that can have both of the following values</dt>
  <dd>One value of the name</dd>
  <dd>Another value of the name</dd>
</dl>
```

Which would render as follows:

```
Name
  Value that applies to the preceding name
```

Name Name

```
 Value that applies to both of the preceding names
```

Name that can have one of the following values

```
 One value of the name
 Another value of the name</pre>
```

Choosing between list types

When trying to decide what type of list to use, you can usually decide by asking two simple questions:

1. Am I defining terms or associating other name/value pairs?
    - If yes, use a description list.
    - If no, don't use a description list — go on to the next question.
2.
3. Is the order of the list items important?
    - If yes, use an ordered list.
    - If no, use an unordered list.
4.

The difference between HTML lists and text

You may be wondering what the difference is between an HTML list and some text with bullets or numbers written in by hand. Well, there are several advantages to using an HTML list:

- If you have to change the order of the list items in an ordered list, you simply move around the list item elements. If you wrote the numbers in manually you would have to go through and change every single item's number to correct the order — which is tedious to say the least!
- Using an HTML list allows you to style the list properly - you can use CSS to style just the list elements. If you just use a blob of text, you will find it more difficult to style the individual items in any useful manner, as the elements used will be the same as used for every other piece of text.
- Using an HTML list gives the content the proper semantic structure, as well as a "list-ish" visual effect. This has important benefits such as allowing screen readers to tell users with visual impairments they are reading a list, rather than just reading out a confusing jumble of text and numbers.

To put it another way: **text and lists are not the same**. Using text instead of a list makes more work for you and can create problems for your document's readers. So if your document needs a list, you should use the correct HTML list.

Nesting lists

A list item can contain another entire list — this is known as "nesting" a list. It is useful for things like tables of contents, such as the one at the start of this article:

1. Chapter One
    1. Section One
    2. Section Two
    3. Section Three
2.
3. Chapter Two
4. Chapter Three

The key to nesting lists is to remember that the nested list should relate to one specific list item. To reflect that in the code, the nested list is contained inside that list item. The code for the list above looks something like this:

```
<code><ol>
```

```
  <li>Chapter One
    <ol>
      <li>Section One</li>
      <li>Section Two </li>
      <li>Section Three </li>
    </ol>
  </li>
  <li>Chapter Two</li>
  <li>Chapter Three  </li>
</ol></code>
```

Note how the nested list starts after the `<li>` and the text of the containing list item ("Chapter One"); then ends before the `</li>` of the containing list item. Nested lists often form the basis for website navigation menus, as they are a good way to define the hierarchical structure of the website.

Theoretically you can nest as many lists as you like, although in practice it can become confusing to nest lists too deeply. For very large lists, you may be better off splitting the content up into several lists with headings instead, or even splitting it up into separate pages.

Step by step example

Let's run through a step by step example, to put all of this together. Consider the following scenario:

You are creating a small website for the HTML Cooking School. On the main page, you are to show a list of categorised recipes, linking through to recipe pages. Each recipe page lists the ingredients required, notes on those ingredients and the preparation method. The three categories are:

- Cakes (including recipes for Plain Sponge, Chocolate Cake and Apple Tea Cake)
- Biscuits (including recipes for ANZAC Biscuits, Jam Drops and Melting Moments)
- Quickbreads (including recipes for Damper and Scones)

The client doesn't mind what order the categories and recipes are shown; they just want to be sure people know which items are categories and which ones are recipes.

Let's step through the process of creating this site.

Main page markup

1. Create a properly-formed HTML page, including a doctype, `<html>`, `<head>` and `<body>` elements, and save it as *stepbystep-main.html*. Add a main heading (`<h1>`) of "HTML Cooking School", and a subheading (`<h2>`) of "Recipes":

```
<h1>HTML Cooking School</h1>
<h2>Recipes</h2>
```

1. You have three categories of recipe to represent, and the order is not important — an unordered list is most appropriate for these, so add the following to your page:

```
<h2>Recipes</h2>
   <ul>
   <li>Cakes</li>
   <li>Biscuits</li>
   <li>Quickbreads</li>
 </ul>
```

Indenting the `<li>` elements makes the code more readable, but it is not required.

1. Now you need to add the recipes as sub-items, for example "Plain Sponge", "Chocolate Cake" and

"Apple Tea Cake" are all part of the "Cakes" category. To do this, you need to create a nested list within each item. Since the order is not important, once again unordered lists are appropriate. To make things easier for the tutorial, I'll get you to link all of the recipes to one single recipe page (html links lets build a web explains HTML links in depth].):

```
<h2>Recipes</h2>
 <ul>
   <li>Cakes
     <ul>
       <li><a href="stepbystep-recipe.html">Plain Sponge</a></li>
       <li><a href="stepbystep-recipe.html">Chocolate Cake</a></li>
       <li><a href="stepbystep-recipe.html">Apple Tea Cake</a></li>
     </ul>
   </li>
   <li>Biscuits
     <ul>
       <li><a href="stepbystep-recipe.html">ANZAC Biscuits</a></li>
       <li><a href="stepbystep-recipe.html">Jam Drops</a></li>
       <li><a href="stepbystep-recipe.html">Melting Moments</a></li>
     </ul>
   </li>
   <li>Breads/quickbreads
     <ul>
       <li><a href="stepbystep-recipe.html">Damper</a></li>
       <li><a href="stepbystep-recipe.html">Scones</a></li>
     </ul>
   </li>
 </ul>
```

The final result should be something similar to Figure 1:

HTML Cooking School

Recipes

Since this is an HTML tutorial, all links go to the sponge recipe. Just thought I'd warn you.

- Cakes
  - Plain Sponge
  - Chocolate Cake
  - Apple Tea Cake
- Biscuits
  - ANZAC Biscuits
  - Jam Drops
  - Melting Moments
- Breads/quickbreads
  - Damper
  - Scones

Figure 1: The finished main page.

You can also view the live example page here (http://dev.opera.com/articles/view/16-html-lists/stepbystep-main.html) .

The recipe page

For the sake of the example, we will just create the sponge cake recipe page — feel free to create the others yourself, using this one as a template. The HTML Cooking School has supplied the sponge recipe to you in a text file, looking like this:

```
Simple Sponge Cake
```

```
Ingredients
3 eggs
100g castor sugar
85g self-raising flour

Notes on ingredients:
  Caster Sugar - Finely granulated white sugar.
  Self-raising flour - A pre-mixed combination of flour and leavening agents (usually salt and baking powder).<

Method
  1. Preheat the oven to 190°C.
  2. Grease a 20cm round cake pan.
  3. In a medium bowl, whip together the eggs and castor sugar until fluffy.
  4. Fold in flour.
  5. Pour mixture into the prepared pan.
  6. Bake for 20 minutes in the preheated oven, or until the top of the cake springs back when lightly pressed.
  7. Cool in the pan over a wire rack.
```

Recipe page markup

1. Create another properly-formed HTML document, and save it as stepbystep-recipe.html. Add the following headings to it:

```
<h1>Simple Sponge Cake</h1>
 <h2>Ingredients</h2>
 <h3>Notes on ingredients</h3>
 <h2>Method</h2>
```

1. The ingredients list has several items but the order isn't important. An unordered list therefore makes sense. Add the following into the `<body>` of your HTML:

```
<h2>Ingredients</h2>
 <ul>
   <li>3 eggs</li>
   <li>100g castor sugar</li>
   <li>85g self-raising flour</li>
 </ul>
```

1. The notes on the ingredients are there to properly define what some of the ingredients are. You need to associate the ingredient — the name — with its value. A description list is right for this purpose. Add the following to your HTML, below the unordered list in the previous step:

```
<h3>Notes on ingredients</h3>
 <dl>
   <dt>Castor Sugar</dt>
     <dd>Finely granulated white sugar.</dd>
   <dt>Self-raising flour</dt>
     <dd>A pre-mixed combination of flour and leavening agents (usually salt and baking powder).</dd>
 </dl>
```

1. The method must obviously follow a single correct order, so it should be an ordered list — add the following to your HTML, below the description list:

```
<h2>Method</h2>
 <ol>
   <li>Preheat the oven to 190°C.</li>
   <li>Grease a 20cm round cake pan.</li>
   <li>In a medium bowl, whip together the eggs and castor sugar until fluffy.</li>
   <li>Fold in flour.</li>
   <li>Pour mixture into the prepared pan.</li>
   <li>Bake for 20 minutes in the preheated oven, or until the top of the cake springs back when lightly presse
   <li>Cool in the pan over a wire rack.</li>
 </ol>
```

The page should look something like Figure 2:

Simple Sponge Cake

Ingredients

- 3 eggs
- 100g castor sugar
- 85g self-raising flour

Notes on ingredients

Castor Sugar
: Finely granulated white sugar.

Self-raising flour
: A pre-mixed combination of flour and leavening agents (usually salt and baking powder).

Method

1. Preheat the oven to 190°C.
2. Grease a 20cm round cake pan.
3. In a medium bowl, whip together the eggs and castor sugar until fluffy.
4. Fold in flour.
5. Pour mixture into the prepared pan.
6. Bake for 20 minutes in the preheated oven, or until the top of the cake springs back when lightly pressed.
7. Cool in the pan over a wire rack.

Figure 2: The finished recipe page.

You can also view the live example page here (http://dev.opera.com/articles/view/16-html-lists/stepbystep-recipe.html) .

You're done!

Summary

By this stage you should have a clear understanding of the three different list types in HTML. Using the step-by-step example, you should have created all three and learned how to nest lists inside list items.

Once you know how to use HTML lists properly, you will probably discover that you use them all the time. There is a lot of content on the web that should have been placed into a list, but was just thrown into a generic element with some line break tags. It's a lazy practice that causes far more problems than it solves — so don't do it! You should always create semantically correct lists to help people read your websites. It is a better practice for everyone, not least yourself when you need to maintain your sites later on.

Further reading

- A List Apart: Taming Lists (http://www.alistapart.com/articles/taminglists/)
- W3C CSS2: list-style-type definition (http://www.w3.org/TR/REC-CSS2/generate.html#lists)

Exercise questions

- What are the three types of HTML list?
- When would you use each type of list? How would you choose between them?
- How do you nest lists?

- Why should you use CSS rather than HTML to style your lists?

Note: This material was originally published as part of the Opera Web Standards Curriculum, available as 16: HTML lists (http://dev.opera.com/articles/view/16-html-lists/) , written by Ben Buchanan. Like the original, it is published under the Creative Commons Attribution, Non Commercial - Share Alike 2.5 (http://creativecommons.org/licenses/by-nc-sa/2.5/) license.

Retrieved from "http://www.w3.org/community/webed/wiki/HTML_lists"
Categories: Tutorials | WSC | HTML

- This page was last modified on 28 January 2012, at 22:45.