Literate sample

Literate sample

This file demonstrates how to write Markdown document with embedded F# snippets that can be transformed into nice HTML using the `literate.fsx` script from the F# Formatting package.

In this case, the document itself is a valid Markdown and you can use standard Markdown features to format the text:

- Here is an example of unordered list and...
- Text formatting including **bold** and *emphasis*

For more information, see the Markdown reference.

Writing F# code

In standard Markdown, you can include code snippets by writing a block indented by four spaces and the code snippet will be turned into a `<pre>` element. If you do the same using Literate F# tool, the code is turned into a nicely formatted F# snippet:

```
1:  /// The Hello World of functional languages!
2:  let rec factorial x =
3:    if x = 0 then 1
4:    else x * (factorial (x - 1))
5:
6:  let f10 = factorial 10
```

Hiding code

If you want to include some code in the source code, but omit it from the output, you can use the `hide` command. You can also use `module=...` to specify that the snippet should be placed in a separate module (e.g. to avoid duplicate definitions).

The value will be deffined in the F# code that is processed and so you can use it from other (visible) code and get correct tool tips:

```
1:  let answer = Hidden.answer
```

Including other snippets

When writing literate programs as Markdown documents, you can also include snippets in other languages. These will not be colorized and processed as F# code samples:

```
Console.WriteLine("Hello world!");
```

This snippet is turned into a `pre` element with the `lang` attribute set to `csharp`.