MATLAB Mapping for Constants

Previous Next

Here are the sample constant definitions once more:

Slice

```
const bool      AppendByDefault = true;
const byte      LowerNibble = 0x0f;
const string    Advice = "Don't Panic!";
const short     TheAnswer = 42;
const double    PI = 3.1416;

enum Fruit { Apple, Pear, Orange }
const Fruit     FavoriteFruit = Pear;
```

Here are the generated definitions for these constants:

MATLAB

```
classdef AppendByDefault
    properties(Constant)
        value logical = true
    end
end

classdef LowerNibble
    properties(Constant)
        value uint8 = 15
    end
end

classdef Advice
    properties(Constant)
        value char = 'Don''t Panic!'
    end
end

classdef TheAnswer
    properties(Constant)
        value int16 = 42
    end
end

classdef PI
    properties(Constant)
        value double = 3.1416
    end
end

classdef FavoriteFruit
    properties(Constant)
        value = Fruit.Pear
    end
end
```

As you can see, each Slice constant is mapped to a MATLAB class with the same name as the constant. The class contains a constant property named `value` that holds the value of the constant.

Slice string literals that contain non-ASCII characters or universal character names are mapped to MATLAB string literals with UTF-16 character codes. For example:

Slice

```
const string Egg = "œuf";
const string Heart = "c\u0153ur";
const string Banana = "\U0001F34C";
```

is mapped to:

MATLAB

```
classdef Egg
    properties(Constant)
        value char = sprintf('\x0153uf')
    end
end

classdef Heart
    properties(Constant)
        value char = sprintf('c\x0153ur')
    end
end

classdef Banana
    properties(Constant)
        value char = sprintf('\xd83c\xdf4c')
    end
end
```

The mapping uses the `sprintf` function to convert encoded strings into native MATLAB character arrays.

Back to Top ^

See Also

- Constants and Literals

Previous Next