Java Compat 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:

Java Compat

```
public interface AppendByDefault
{
    boolean value = true;
}

public interface LowerNibble
{
    byte value = 15;
}

public interface Advice
{
    String value = "Don't Panic!";
}

public interface TheAnswer
{
    short value = 42;
}

public interface PI
{
    double value = 3.1416;
}

public interface FavoriteFruit
{
    Fruit value = Fruit.Pear;
}
```

As you can see, each Slice constant is mapped to a Java interface with the same name as the constant. The interface contains a member named `value` th at holds the value of the constant.

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

Slice

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

is mapped to:

Java Compat

```
public interface Egg
{
    String value = "\u0153uf";
}

public interface Heart
{
    String value = "c\u0153ur";
}

public interface Banana
{
    String value = "\ud83c\udf4c";
}
```

Back to Top ^

See Also

- Constants and Literals

Previous Next