9 Enums

An enum type is a distinct subtype of the Number primitive type with an associated set of named constants that define the possible values of the enum type.

9.1 Enum Declarations

An enum declaration declares an ***enum type*** and an ***enum object*** in the containing module.

EnumDeclaration:
`enum` *Identifier* `{` $EnumBody_{opt}$ `}`

The enum type and enum object declared by an *EnumDeclaration* both have the name given by the *Identifier* of the declaration. The enum type is a distinct subtype of the Number primitive type. The enum object is a variable of an anonymous object type containing a set of properties, all of the enum type, corresponding to the values declared for the enum type in the body of the declaration. The enum object's type furthermore includes a numeric index signature with the signature '[x: number]: string'.

The *Identifier* of an enum declaration may not be one of the predefined type names (section 3.6.1).

The example

```
enum Color { Red, Green, Blue }
```

declares a subtype of the Number primitive type called 'Color' and introduces a variable 'Color' with a type that corresponds to the declaration

```
var Color: {
    [x: number]: string;
    Red: Color;
    Green: Color;
    Blue: Color;
};
```

The numeric index signature reflects a "reverse mapping" that is automatically generated in every enum object, as described in section 9.4. The reverse mapping provides a convenient way to obtain the string representation of an enum value. For example

```
var c = Color.Red;
console.log(Color[c]);  // Outputs "Red"
```

9.2 Enum Members

The body of an enum declaration defines zero or more enum members which are the named values of the enum type. Each enum member has an associated numeric value of the primitive type introduced by the enum declaration.