Notes - 12/5

Wednesday, December 5, 2012 12:49 PM

[In 41/1753 at 1:00 today.]

Agenda:

- Generics update
    - Maybe nothing new for today?
- Compile-on-save and compilation configuration
    - Notes here: Compile-on-save and compiler settings configuration (Web view)
- Enums

    - Previous viewpoint:
        - Enums should become a compile-time only thing
    - Philosophy on our compilation pipeline
        - Our compiler really does .ts -> .js
        - Then other steps (separate tools) do:
            - Concatenation
            - Minification
            - Optimization (inling, and dead code elimination)
            - Dependency analysis
            - R.js
            - Etc.
        - There are some other pre-process pieces
    - Current thinking:
        - Enums are manifest
        - We want reverse mapping
            - Some issues with numbers, but okay to be somewhat unpredictable here

```
Classes:
        Foo: number = 3; (this is good)
Interfaces:
        Foo: string;
Enums
        Foo = 3,
Object
        Foo: 3,
```

```
enum Color { "red", "blue" }
enum Day { Monday: 0, Tuesday: 1, "Wed nes day": 3}

enum Day { Monday: 0, Tuesday: 1, "Wed nes day": 3}

var Day = {
  "Monday": 0, 0: "Monday",
  "Tuesday": 1, 1: "Tuesday"
}

var Day = {
  Monday: 0,
  Tuesday: 1
  0: "Monday",
  1: "Tuesday
}
```

```
/* WE DO THIS */

enum Day { Monday: 0, Tuesday: 1, "Wed nes day": Day.Monday | Day.Tuesday,
}
// LHSes can be anything that is a valid Object Literal LHS *except* a number

-->



var Day;
(function(Day) {
  Day[Day.Monday = 0] = "Monday";
  Day[Day.Tuesday = 1] = "Tuesday";
  Day[Day["Wed nes day"] = Day.Monday | Day.Tuesday] = "Wed nes day";
})(Day = {});

// No string enums
// Workitem: design overloading on constants
```

Typing:

- Enums are branded subtypes of number

/* END WE DO THIS */

```
var Day = (function() {
  var Day = {};
  Day[Day.Monday = 0] = "Monday";
  Day[Day.Tuesday = 1] = "Tuesday";
  Day[Day["Wed nes day"] = Day.Monday | Day.Tuesday] = "Wed nes day";
 return Day;
})();


/* WE DO THIS */

enum Day { Monday: 0, Tuesday: 1, "Wed nes day": Day.Monday | Day.Tuesday, }
// LHSes can be anything that is a valid Object Literal LHS *except* a number

-->



var Day;
(function(Day) {
  Day[Day.Monday = 0] = "Monday";
  Day[Day.Tuesday = 1] = "Tuesday";
  Day[Day["Wed nes day"] = Day.Monday | Day.Tuesday] = "Wed nes day";
})(Day = {});

// No string enums
// Workitem: design overloading on constants
```

Typing:

- Enums are branded subtypes of number

```
/* END WE DO THIS */



var Day;
(function(_) {
  _[_.Monday = 0] = "Monday";
  _[_.Tuesday = 1] = "Tuesday";
  _[_["Wed nes day"] = Day.Monday | Day.Tuesday] = "Wed nes day";
})(Day = {});



var Day = (function() {
  var Day = {}
  Day[Day["Monday"] = 0] = "Monday";
  Day[Day["Tuesday"] = 1] = "Tuesday";
})()


enum Day { Monday = 0, Tuesday = 1}

var c: Color = "red"
var d: Day = Day.Monday;


document.createElement("div")

interface GetOptions {
 method: "GET",
 //...
}
interface PutOptions {
 method: "PUT",
  //...
}
$.ajax(options: GetOptions)
$.ajax(options: PutOptions)
$.ajax({method: "GET", ...})

addEventListener('click', function(mev) {
})
// routing - /foo/:user
```

- Update on Fundules, clodules
- Design backlog
  - "self type" in classes (see forum thread @ Oct 8 at 3:22 PM)
  - Bringing back explicit "this" parameter type
  - ... for arguments
  - String interpolation
  - Overloading on constants
  - Rubber stamp enum update
  - Type-only mixins
  - Decorations
  - Static initialization (#74)
  - Thinking on how to approach async via generators (#38)
  - SkyDrive feedback – implications for future thinking?
- Others?
-