Scripting rules of thumb

JavaScript is case sensitive.
Identifiers as traditional, except can be unicode and can begin with a $.
  Avoid $ as it is commonly used in external frameworks (JQuery)
Camel casing is traditional.
  salesTaxRate, firstName
Beware that existing JavaScript components inconsistent (p. xxi).
Comments:
  // for single line
  /* */ for multi-line
Scripting rules of thumb

Statements should be terminated with a semicolon, even though parser will automatically put in if omitted. Why?

Postel's Law

Code blocks use {}. Conventional in JavaScript to use even around single-line blocks (if statements) and to use Java-style formatting.

```javascript
if (condition) {
    do this;
}
```

Opening (or closing) brace goes on line with the statement that controls access to the code block.

Variables and typing

Variables are not explicitly typed, and type may be changed implicitly through assignment.

'var' placed in front of identifier to denote a variable declaration.

```javascript
var name;
var age = 17;
var gender, grade;
var shoeSize = 7, hairColor = "red";
```
Variable scoping rules.

Variable created with var--local scope (same code block) only.

Variables *can* be created without var (appear in assignment). Those variables have *global scope*.

- Hard to maintain.
- May cause name collision
- Don't do this!

Identifier types

6 JavaScript identifier types:
- undefined (variable declared, not initialized)
- boolean—true or false
- string
- number—no distinction between integers and floats
  - [http://einstein.etsu.edu/~pittaes/CSCI3110/examples/2-1.htm](http://einstein.etsu.edu/~pittaes/CSCI3110/examples/2-1.htm)
- object
- function

*typeof* operator allows us to determine type of an in-scope identifier.

- [http://einstein.etsu.edu/~pittaes/CSCI3110/examples/2-2.htm](http://einstein.etsu.edu/~pittaes/CSCI3110/examples/2-2.htm)
Type conversion functions

**Boolean(val)**—evaluates val and returns true or false.
  - true: boolean true, nonempty string, nonzero number, any object
  - false: boolean false, empty string, 0, NaN, null, undefined.

**Number(val)**—performs numeric conversion on val
  - boolean true returns 1, false returns 0.
  - if val is numeric, returns val.
  - if val is undefined, returns NaN.
  - null string returns 0.
  - String containing a number ("123"), returns that number.
  - String containing text, returns NaN.

**parseInt(val)**—returns val converted to an "integer"
  - If first non-whitespace character in val is not numeric or string is null, returns NaN.
  - Continues parsing val until reaches end or a non-numeric value. Returns that outcome.
  - If numeric substring begins 0x, interpreted as hexadecimal. If begins with 0, interpreted as octal.
  - Favored over Number since returns NaN on a null string val instead of 0.
  - An optional second parameter can specify the base of val. Returned value is always base 10.
  - [http://einstein.etsu.edu/~pittares/CSCI3110/examples/2-3.htm](http://einstein.etsu.edu/~pittares/CSCI3110/examples/2-3.htm)
Type conversion functions

`parseFloat(val)`—returns `val` converted to a "float"

  Similar to `parseInt`, except
  . is a valid part of a number, but can only occur once.
  Zeros at the beginning are ignored.
  If `val` in hexadecimal form ("0x...") 0 is returned.
  If `val` in scientific notation form, float equivalent is returned. (2.7845e10)

`val.toString()` returns `val` as a string.

  If `val` is null or undefined, "null" or "undefined" returned.
  If `val` is number, optional parameter can specify base of output.
  var `x` = 10;
  var `y` = `x`.toString(2);

In class practice
Copyrights

Presentation prepared by and copyright of Dr. Tony Pittarese, East Tennessee State University, Computer and Information Sciences Dept. (pittares@etsu.edu)

Podcast lecture related to this presentation available via ETSU iTunesU.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System e, System p, System z, System z9, System z10, System z196, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VMS, i5/OS, i5/OS, z/OS, z/OS(900), z900, 422/400, 422/440, 422/830 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BladeCenter, System Storage, G/OPS, HACMP, Tivoli, DB2 Connect, IAF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, Websphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Oracle is a registered trademark of Oracle Corporation.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, XI Professional, and other Business Objects products and services mentioned herein, as well as their respective logos are trademarks or registered trademarks of Business Objects S.A. in the United States and in other countries. Business Objects is an SAP company.

ERPsim is a registered copyright of ERPsim Labs, HEC Montreal.

Other products mentioned in this presentation are trademarks of their respective owners.