


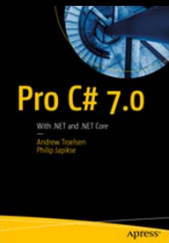

JavaScript for the C# Developer

Philip Japikse (@skimedic)
www.skimedic.com/blog
Consultant, Teacher, Writer



Phil.About()

- Consultant, Coach, Author, Teacher
 - Lynda.com (<http://bit.ly/skimediclyndacourses>)
 - Apress.com (<http://bit.ly/apressbooks>)
- Microsoft MVP, ASPInsider, MCSD, MCDBA, CSM, CSP
- Founder, Agile Conferences, Inc.
 - <http://www.dayofagile.org>
- President, Cincinnati .NET User's Group



All slides copyright Philip Japikse <http://www.skimedic.com>

JAVASCRIPT GRAMMAR

All slides copyright Philip Japikse <http://www.skimedic.com>

GRAMMAR IS SIMILAR TO C#

- Case Sensitive (camel case is convention)
- Comments (`//` or `/* */`)
- White space is not significant
- Variables start with character
- Strings
 - Can use Single or Double Quotes (must match)
 - Escape characters with backslash
- Statement terminator (`;`)
 - Be explicit!
- Place open brace on same line
- "use strict"
 - "Error: Variable undefined in strict mode"

All slides copyright Philip Japikse <http://www.skimedic.com>

OPERATORS AND DATA TYPES

All slides copyright Philip Japikse <http://www.skimedic.com>

OPERATORS

- Standard Order of Operations applies
 - Parenthetical groupings
 - Exponents/Roots (`Math.pow(x,y)` || `XY`, `Math.sqrt(x)`)
 - Multiplication/Division/Remainder (`*`, `/`, `%`)
 - Addition/Subtraction (`+`, `-`)
- Unary
 - Type/To Number/Negation/Logical Not (`typeof`, `+`, `-`, `!`)
- Ternary
 - `(boolean) ? DolfTrue : DolfFalse;`

All slides copyright Philip Japikse <http://www.skimedic.com>

OPERATORS

- Logical And/Logical Or (&,&|)
- Inequality
 - <, <=, >=, >
- Equality
 - With type conversion (==,!=)
 - ("1" == 1) //true
 - Without type conversion (===, !==)
 - ("1" === 1) //false

All slides copyright Philip Japikse <http://www.skimedic.com>

CORE DATA TYPES AND VALUES

Data Types

- String
- Number
 - Double precision 64-bit binary
- Boolean
- Array
- Date
- RegEx
- Function
- Object

Built in Values

- Boolean
 - true, false
- null
- undefined
- NaN
 - isNaN("foo") //true
- Infinity
- 10/0

All slides copyright Philip Japikse <http://www.skimedic.com>

STATEMENTS

```
if (condition) { /* DoIfTrue; */ } else { /* DoIfFalse;*/ }

while (false) { /* do work */ }

do { /* work */ } while (false);

for (var x=0;x<10;x++) { /* do work */}

var arr = [1,2]; for (var key in arr) { /* do something */ }

try {} catch (ex) {}

switch (x) {
  case 1: /* do something */ break;
  case 2: /* falls through */
  case 3: break;
  default:
```

All slides copyright Philip Japikse <http://www.skimedic.com>

DEMO DEWO

Data Types

Truthiness

All slides copyright Philip Japikse <http://www.skimedic.com>

ARRAYS

All slides copyright Philip Japikse <http://www.skimedic.com>

ARRAYS

- Extremely useful
- Loosely typed
 - Indices can be strings
- Properties
 - Length = highest index + 1
- "Standard Operators"
 - indexOf/lastIndexOf
 - sort(function)
 - reverse

All slides copyright Philip Japikse <http://www.skimedic.com>

ARRAY METHODS

- `forEach(function(index,value)`
 - Executes function for each element
- `every(functionTest(value,index)`
 - True if all elements match test
- `some(functionTest(value,index)`
 - True if one element matches test
- `filter(functionTest(value,index)`
 - New array where elements match test
- `join([separator])`
 - Create string from all values
- `map(function(value,index)`
 - Creates a new array from return value of the function
- `reduce[Right](function(previousValue,currentValue,index) [, initialValue])`
 - Recursively process the elements

All slides copyright Philip Japikse <http://www.skimedic.com>

ARRAY METHODS

- `pop`
 - Remove and return last element
- `shift`
 - Remove and return first element
- `push([items])`
 - Add elements to end and return length
- `unShift([items])`
 - Add elements at start and return length
- `slice(start_pos,length)`
 - Returns new array
- `splice(start_pos,length,[items])`
 - Remove items (length != 0), Adds [items] at start_pos

All slides copyright Philip Japikse <http://www.skimedic.com>

DEMO DEMO

Array Examples

All slides copyright Philip Japikse <http://www.skimedic.com>

FUNCTIONS

All slides copyright Philip Japikse <http://www.skimedic.com>

FUNCTIONS

- Functions in JavaScript are first class objects
- Can be named or anonymous
- Can be passed as arguments to other functions
- All arguments are optional
- Additional arguments can be passed in
 - Accessed through the arguments collection

All slides copyright Philip Japikse <http://www.skimedic.com>

IMMEDIATELY INVOKED FUNCTION EXPRESSIONS (IIFE)

- Used to ensure all necessary code is executed on load
- Creates private scope of included variables
- Default pattern in most libraries

All slides copyright Philip Japikse <http://www.skimedic.com>

DEMO DEMO

Functions

Function Parameters

Self Executing Functions

All slides copyright Philip Japikse <http://www.skimedic.com>

OBJECTS

All slides copyright Philip Japikse <http://www.skimedic.com>

SIMPLE OBJECTS

- Create simple objects with name/value pair (similar to JSON)
- JavaScript is Dynamic
 - Properties can be added at anytime
 - Properties can be removed via "delete"
 - Validate existence with `hasOwnProperty()`
- Accessed through "dot" notation or brackets
- Objects can be nested
- Properties can be functions

All slides copyright Philip Japikse <http://www.skimedic.com>

CUSTOM TYPES

- All features of simple objects
- Created using a constructor function
 - Create new instances using "new"
- Access/Add shared properties through object's prototype
 - Creates pseudo inheritance (copy on write)
- Can use `Object.Create` to lock down prototypes
- Can have static members

All slides copyright Philip Japikse <http://www.skimedic.com>

DEMO DEMO

Objects

All slides copyright Philip Japikse <http://www.skimedic.com>

SCOPE & NAMESPACES

All slides copyright Philip Japikse <http://www.skimedic.com>

SCOPE

- Only two options – Global or Function
- Blocks don't encapsulate variables
- Order doesn't matter
 - As long as they are declared, variables get hoisted
- Can (and should) force explicit scoping
 - "use strict"

All slides copyright Philip Japikse <http://www.skimedic.com>

NAMESPACES

- Encapsulate Variables
 - Much like C#, VB.NET
- Helps prevent collisions with other frameworks
- Leverage dynamic nature of JavaScript

All slides copyright Philip Japikse <http://www.skimedic.com>

DEMO DEMO

Scope and Hoisting
Namespaces

All slides copyright Philip Japikse <http://www.skimedic.com>

CLOSURES

All slides copyright Philip Japikse <http://www.skimedic.com>

CLOSURES

- Local variables for a function kept alive after the function has returned
 - Created by using a function inside of a function
- Internal function can reference local variables inside returned function
 - In C#, this would have been destroyed
- A nice little tutorial:
 - <http://www.javascriptkit.com/javatutors/closures.shtml>

All slides copyright Philip Japikse <http://www.skimedic.com>

DEMO DEWO

Closures, Memoization

All slides copyright Philip Japikse <http://www.skimedic.com>

JSON

All slides copyright Philip Japikse <http://www.skimedic.com>

JSON SERIALIZATION

- JavaScript Object Notation
 - Largely replaced POX and SOAP
- Converts object graph to string
 - `JSON.stringify(myobject)`
- Converts string to object graph
 - `JSON.parse(text [,reviver]);`
 - `JSON.parse('{ "firstName": "Philip", "lastName": "Japikse" })'`

All slides copyright Philip Japikse <http://www.skimedic.com>

DEMO DEMO

JSON

All slides copyright Philip Japikse <http://www.skimedic.com>

PROMISES

All slides copyright Philip Japikse <http://www.skimedic.com>

PROMISES

- Not yet part of the standard
 - Implemented by JQuery, WinJS, NodeJS
- Similar to await/async
 - Does not necessarily mean multi threaded

All slides copyright Philip Japikse <http://www.skimedic.com>

PROMISES, PROMISES, PROMISES

```
Promise
  .then(onComplete, onError, onProgress)
  .done(onComplete, onError, onProgress);
-----
return new WinJS.Promise(
  function(fDone, fError, fProgress){
    var contacts = JSON.stringify(contactList.slice(0));
    FileHandling.saveContactsFile(contacts)
      .then(function () { fDone("Saved"); });
  }
);
```

All slides copyright Philip Japikse <http://www.skimedic.com>

Contact Me

skimedic@outlook.com
www.skimedic.com/blog
www.twitter.com/skimedic

<http://bit.ly/skimediclyndacourses>
<http://bit.ly/apressbooks>

www.hallwayconversations.com

Thank
You!

All slides copyright Philip Japikse <http://www.skimedic.com>