Swift Syntax compared with Objective-C, C# and JavaScript

	Swift	Objective-C	C#	JavaScript
Variables				
	`var number = 1` `var number: Int = 1`	`int number = 1;`	`var number = 1;` `int number = 1;`	var number = 1;
	`var language = "Swift"` `var language: String = "Swift"`	`NSString *language = @"Swift";`	`var language = "Swift";` `string language = "Swift";`	var language = “Swift";
Constants				
	`let language = "Swift"` `let language: String = "Swift"`	`NSString *language = @"Swift";`	`const string language = "Swift";`	Not Applicable
Arrays				
Declare	`var arr = ["first" , "second"]`	`NSArray *arr = @[@"first", @"second"];`	`var arr = new[] { "first", "second" };`	var arr = [“first” , “second”];
Get	`var order = arr[0]`	`NSString *order = arr[0];`	`var order = arr[0];`	var order = arr[0];
Set	`arr[0] = "zero"`	`[arr replaceObjectAtIndex:0 withObject: @"third"];`	`arr[0] = "third";`	arr[0] = “third”;

Append	`arr += "fourth"`
	`arr.append("fifth")`
	`arr += ["sixth", "seventh"]`
		`[arr addObject:  @"fourth"];`	`Array.Resize(ref arr, arr.Length + 1);`
			`arr[arr.Length - 1] = "Three";`
				arr.push("fourth");

Enumerate	`for item in arr{`
	`  //do something`
	`}`
		`for(NSString *item in arr)`
		`{`
		`  // do something`
		`}`
			`foreach (var item in arr)`
			`{`
			`  // do something`
			`}`	

Dictionaries				

Declare	`var dict = Dictionary<String, String>()`
	
	`var dict = ["MEL": "Melbourne", "SYD": "Sydney"]`
		`NSDictionary *dict = @{`
		`@"MEL" : @"Melbourne",`
		`@"SYD" : @"Sydney"`
		`};`
			`var dict = new Dictionary<string, string>`
			`{`
			`  { "MEL", "Melbourne" },`
			`  { "SYD", "Sydney" }`
			`};`	

Get	`var entry = dict["MEL"]`	`NSString *entry = dict[@"MEL"];`	`var entry = dict["MEL"];`	
Set	`dict["PER"] = "Perth"`	`dict[@"PER"] = @"Perth";`	`dict["PER"] = "Perth";`	
Append	As above	As above	As above	

Enumerate	`for (cityCode, cityName) in dict {`
	`println("\(cityCode): \(cityName)")`
	`}`
		`for (id key in dict) {`
		` NSLog(@"key: %@, value: %@", key, dict[key]);`
		`}`
			`foreach (var item in dict)`
			`{`
			`  var cityCode = item.Key;`
			`  var cityName = item.Value;`
			`}`	

Loops				

for	`for var number = 1; number < 5; number++ {`
	`  //do something`
	`}`
		`for (int number = 1; number <5; number++)`
		`{`
		`  //do something`
		`}`
			`for (int number = 1; number < 5; number++)`
			`{`
			`  //do something`
			`}`	

For in	`for city in arr {`
	`  println(city)`
	`}`
		`for (id city in arr)`
		`  // do something`
		`}`
			`foreach (var city in arr)`
			`{`
			`  // do something`
			`}`	

while	`var number = 1` `while number < 10 {`	`int number = 1;` `while (number <10) {`	`int number = 1;` `while (number < 10)`	

	println(number)
	number++
	}
		NSLog(@"%i", number);
		number++;
		}
			{
			Console.WriteLine(number);
			number++;
			}	

Do while	var number = 9
	do {
	println(number)
	number++
	}while number < 10
		int number = 9;
		do{
		NSLog(@"%i", number);
		number ++;
		} while (number < 10);
			int number = 9;
			
			do {
			Console.WriteLine(number);
			number++;
			} while (number < 10);	

Conditionals				

If	if city == "MEL"{
	println("Melbourne")
	}
		if ([city isEqualToString:
		@"MEL"]) {
		NSLog(@"Melbourne");
		}
			if (city == "MEL")
			{
			Console.WriteLine("Melbourne");
			}	

If - else	if city == "MEL"{
	println("Melbourne")
	}else if city == "SYD" {
	println("Sydney")
	}else {
	println("Perth")
	}
		if ([city isEqualToString:
		@"MEL"]) {
		NSLog(@"Melbourne");
		}
		else if ([city isEqualToString:
		@"SYD"])
		{
		NSLog(@"Sydney");
		}
		else
		}
		NSLog(@"Perth");
		}
			if (city == "MEL")
			{
			Console.WriteLine("Melbourne");
			}
			else if (city == "SYD")
			{
			Console.WriteLine("Sydney");
			}
			else
			{
			Console.WriteLine("Perth");
			}	

Switch	switch city {
	case "MEL":
	println("Melbourne")
	case "SYD":
	println("Sydney")
	default:
	println("Perth")
	}
		int number =2;
		
		switch (number)
		{
		case 1:
		NSLog (@"one");
		break;
		
		case 2:
		NSLog (@"two");
		break;
		
		default:
		NSLog (@"unknown");
		break;
		}
			switch (city)
			{
			case "MEL":
			Console.WriteLine("Melbourne");
			break;
			
			case "SYD":
			Console.WriteLine("Sydney");
			break;
			
			default:
			Console.WriteLine("Perth");
			break;
			}	

Functions				

Declare	func sayName(){
	println("Patrick")
	}
		void sayName()
		{
		NSLog(@"Patrick");
		}
			public void SayName()
			{
			Console.WriteLine("Patrick");
			}	

Single Parameter	func sayName(name: String){
	println(name)
	}
		void sayName (NSString *name)
		{
		NSLog(@"%@", name);
			public void SayName(String name)
			{
			Console.WriteLine(name);
			}	

		}		

Multiple Parameters	func sayName(name: String, lastName: String){
	    println("\(name) \(lastName)")
	}
		void sayTwoNames (NSString *name, NSString *lastName)
		{
		    NSLog(@"%@ %@", name, lastName);
		}
			public void SayName(String name, String lastName)
			{
			    Console.WriteLine(name + lastName);
			}	

Return Value	func sayName(name: String, lastName: String) ->String {
	    return "\(name) \(lastName)"
	
	}
		NSString *saymyname(NSString *name, NSString *lastname) {
		    NSString *fullName = [NSString stringWithFormat:@"%@ %@", name, lastname];
		    return fullName;
		}
			public String SayName(String name, String lastName)
			{
			    return (name + lastName);
			}	