You Already know C# 3.0

It’s simpler syntax for techniques you already know and use

Bill Wagner
SRT Solutions
wwagner@srtsolutions.com
July 10, 2007

About Bill Wagner

- Microsoft Regional Director
  - http://www.microsoftregionaldirectors.com/
- C# MVP
- Author of Effective C#
- wwagner@srtsolutions.com
- http://www.srtsolutions.com/public/blog/20574

SRT SOLUTIONS
We Develop Software Developers

Agenda(s)

- Door Number 1
  - Questions?
  - Answers
  - Investigation
- Caveats:
  - About C# / .NET
  - I may say “I don’t know”
  - I may say “let’s find out”

- Door Number 2
  - Start with an Orcas Application
  - Remove C# 3.0 features (back port to 2.0)
  - Understand differences
  - Compare

SRT SOLUTIONS
We Develop Software Developers

Finding Process Statistics

```
var processes = (from process in
   System.Diagnostics.Process.GetProcesses()
      where (process.Threads.Count > 1)
      orderby process.BasePriority descending,
      process.PeakWorkingSet64 descending
         select new
         {
           Name = process.ProcessName,
           Priority = process.BasePriority,
           MaxMemory = process.PeakWorkingSet64
         }
    ).Take(25);
```

- Search all processes
- Must have active threads
- Sort by priority
- Sort by memory
- Grab a few fields
- Only 25 records

SRT SOLUTIONS
We Develop Software Developers

Remove query keywords

```
var processes =
  System.Diagnostics.Process.GetProcesses().
  Where(process => process.Threads.Count > 1).
  OrderByDescending(
    process => process.PeakWorkingSet64).
  OrderByDescending(
    process => process.BasePriority).
  Select(process => new
    {
      Name = process.ProcessName,
      Priority = process.BasePriority,
      MaxMemory = process.PeakWorkingSet64
      })
  .Take(25);
```

- Lost Clarity
  - Order By in wrong order to get the right order
- More Verbose
  - All those .'s

SRT SOLUTIONS
We Develop Software Developers

Removing ‘var’ in N steps

- Create the named class for results
  - C# 3.0 version: implicit properties
  - C# 2.0 version: More typing
  - It’s all grunt work
    - No real thought, or design
- Change the Query
  - Return the new named type

Storage Class 3.0 version

```
public class ProcessStatistics
{
    public string Name
    {
        get;
        set;
    }
    public string Priority
    {
        get;
        set;
    }
    public long MaxMemory
    {
        get;
        set;
    }
}
```

- Implicit Properties
- Compiler Generations
  - Backing field
  - Get, Set, body

SRT SOLUTIONS
We Develop Software Developers

Storage Class 2.0 version

```
public class ProcessStatistics
{
    private readonly string name;
    public string Name
    {
        get { return name; }
    }
    private readonly int priority;
    public int Priority
    {
        get { return priority; }
    }
    private readonly long maxMemory;
    public long MaxMemory
    {
        get { return maxMemory; }
    }

    public ProcessStatistics(string Name, int Priority, long
     MaxMemory)
    {
        name = Name;
        priority = Priority;
        maxMemory = MaxMemory;
    }
}
```

- Much more typing
- No new features
- More maintenance
- Decreased Productivity

SRT SOLUTIONS
We Develop Software Developers

Removing Var from Query

```
IEnumerable<ProcessStatistics> processes =
  Process.GetProcesses().
  Where(process =>
    process.Threads.Count > 1).
  OrderByDescending(
    process => process.PeakWorkingSet64).
  OrderByDescending(process =>
    process.BasePriority).
  Select(process => new ProcessStatistics(
    process.ProcessName,
    process.BasePriority,
    process.PeakWorkingSet64
 )).Take(25);
```

- Select Changes
- Explicit construction
- Property names are lost
  - At least here

SRT SOLUTIONS
We Develop Software Developers

Removing Extension Methods V1

```
IEnumerable<ProcessStatistics> processes =
 Enumerable.Take(
 Enumerable.Select(
 Enumerable.OrderByDescending(
 Enumerable.OrderByDescending(
 Enumerable.Where(
   Process.GetProcesses(),
 process => process.Threads.Count > 1),
 process => process.PeakWorkingSet64),
 process => process.BasePriority),
 process => new ProcessStatistics(
 process.ProcessName,
 process.BasePriority,
 process.PeakWorkingSet64)), 25);
```

- Call Order is reversed
  - Take,
  - Select,
  - Orderby,
  - Where
- Parms read from inward out

SRT SOLUTIONS
We Develop Software Developers

Removing Extension Methods V2

```
IEnumerable<System.Diagnostics.Process>
  whereResult = Enumerable.Where(
    Process.GetProcesses(),
    process => process.Threads.Count > 1);

IEnumerable<System.Diagnostics.Process>
  secondaryOrder =
    Enumerable.OrderByDescending(
    whereResult,
    process => process.PeakWorkingSet64);

IEnumerable<System.Diagnostics.Process>
  primaryOrder =
    Enumerable.OrderByDescending(
    secondaryOrder,
    process => process.BasePriority);
```

```
IEnumerable<ProcessStatistics>
    stats =
     Enumerable.Select(
     primaryOrder,
     process => new
       ProcessStatistics(
       process.ProcessName,
       process.BasePriority,
       process.PeakWorkingSet64));

IEnumerable<ProcessStatistics>
    processes =
     Enumerable.Take(
     stats,
     25);
```

SRT SOLUTIONS
We Develop Software Developers

Critique

- Imperative Programming
  1. Find matching processes
  2. Create Secondary Order
  3. Create Primary Order
  4. Select Output Type
  5. Filter by Top 25
- How, not What

Lambda = Anonymous Delegate

```
IEnumerable<Process> whereResult =
  Enumerable.Where(Process.GetProcesses(),
  delegate(Process process)
    {
       return process.Threads.Count > 1;
    }
);
 // etc.
IEnumerable<ProcessStatistics> stats =
  Enumerable.Select(primaryOrder,
  delegate(Process process)
    {
      return new ProcessStatistics(
        process.ProcessName,
        process.BasePriority,
        process.PeakWorkingSet64);
    }
);
```

- More Verbose syntax
- Full body of method defined
- Same Result

SRT SOLUTIONS
We Develop Software Developers

A Final Comparison

```
var processes = (from process in
  Process.GetProcesses()
  where (process.Threads.Count > 1)
  orderby process.BasePriority descending,
  process.PeakWorkingSet64 descending
  select new
  {
    Name = process.ProcessName,
    Priority = process.BasePriority,
    MaxMemory = process.PeakWorkingSet64
  }
  ).Take(25);
```

```
public class ProcessStatistics
{
    private readonly string name;
    public string Name
    {
        get { return name; }
    }
    private readonly int priority;
    public int Priority
    {
        get { return priority; }
    }
    private readonly long maxMemory;
    public long MaxMemory
    {
        get { return maxMemory; }
    }
    public ProcessStatistics(string Name, int Priority, long MaxMemory)
    {
        name = Name;
        priority = Priority;
        maxMemory = MaxMemory;
    }
}

IEnumerable<System.Diagnostics.Process> whereResult =
    Enumerable.Where(
    System.Diagnostics.Process.GetProcesses(),
    delegate(System.Diagnostics.Process process)
    {
        return process.Threads.Count > 1;
    }
);
IEnumerable<System.Diagnostics.Process> secondaryOrder =
    Enumerable.OrderByDescending(
    whereResult,
    delegate(System.Diagnostics.Process process)
    {
        return process.PeakWorkingSet64;
    }
);
IEnumerable<System.Diagnostics.Process> primaryOrder =
    Enumerable.OrderByDescending(
    secondaryOrder,
    delegate(System.Diagnostics.Process process)
    {
        return process.BasePriority;
    }
);
IEnumerable<ProcessStatistics> stats =
    Enumerable.Select(
    primaryOrder,
    delegate(System.Diagnostics.Process process)
    {
        return new ProcessStatistics(
        process.ProcessName,
        process.BasePriority,
        process.PeakWorkingSet64);
    }
);
IEnumerable<ProcessStatistics> processes =
    Enumerable.Take(
    stats,
    25);
```

SRT SOLUTIONS
We Develop Software Developers

Comparison

- C# 3.0 Advantages
  - Focuses on What code does
    - Declarative
  - More concise, yet more readable
  - Easier to Maintain, more productivity
  - More Powerful
- Caveats:
  - More Power = More Danger

Where you can Learn More

- C# Zone: http://csharp.net
- Orcas Beta: search = “Orcas Beta 1”
- C# Feeds:
  - http://csharpfeeds.com/Default.aspx
- My blog (shameless plug)

SRT SOLUTIONS
We Develop Software Developers

Questions

SRT SOLUTIONS
We Develop Software Developers