**Project Description**

**Industrial-Strength Evolutionary Computation for .NET 4.0**

**This is an independent port of** [**Sean Luke's venerable ECJ project (GMU EC Lab) Version 20**](http://www.cs.gmu.edu/~eclab/projects/ecj/)

**NOTE: This conversion to C# cannot and will not in any way be supported by Sean or any other member of his team at GMU. Please do not even ask! It is incredible that software of this caliber is made available in open-source form, and that is more than enough support!**   
  
It is hard to imagine how many hours Sean and his team have put into this over the course of **13 years**. But suffice it to say:   
**ECJ is an 800 lb Gorilla in the realm of Evolutionary Computation and Stochastic Optimization**.   
  
**MILD WARNING: ECJ** is used for some very serious business (and this C# version has, or will shortly have, all of the same capabilities). *But you don't need to get scared off by that*. A comprehensive collection of samples is provided with **ECJ**, and the documentation that you can browse on **Sean's** site covers all of the basics and well beyond. I highly recommend that newbies work with **ECJ** in **JAVA** before you decide if this converted fork makes sense for you. The need for tight native .NET platform integration is potentially uncalled for, and you must ask yourself, **"Do I really Need This?". Chances are, you probably don't.** But I personally really *DO* need this, and I’m happy to spare others with similar requirements all of the heavy lifting required to get it this far. *Trust me, it is a DAUNTING challenge to preserve delicate algorithms when there are so many moving parts.*   
  
I have made all of the samples bundled in the original Java version equally accessible to those of you who need orientation in C#. Each sample is set up as a bare-bones console app that you can "fiddle" with to your heart's content. But also, and this is something that is *NOT* provided in the original software, there are **over 500 unit tests** that I wrote during the conversion process. This turns out to be quite a useful side-effect of going through the whole torturous exercise in the first place! These tests, although perhaps still a little incoherent in places (because they focus on specific conversion issues), make it even easier to dig deep without stumbling around in total bewilderment (well, you may still stumble around, but they do provide a bit of a handrail to lean on when start feeling sick to your stomach). \'-)   
  
**Side Note: *There are likely to wind up being several thousands of tests associated with this software. I only covered what was causing difficulty during the conversion process, and that is but a small fraction of the full coverage this software deserves. ("sheesh", as Sean is wont to say, I must be freaking crazy!)***   
  
If you are not familiar with the purpose of this software, you can purchase Sean's excellent book entitled [**The Essentials of Metaheuristics**](http://www.lulu.com/product/paperback/essentials-of-metaheuristics/15028803) (or download the electronic version for free). You can also find numerous articles and academic papers on the website listed above, covering a variety of related topics. If you plan on doing really serious work in this field, you might also consider starting out with the early works of [**John Koza (Genetic Programming I, II, et al)**](http://www.genetic-programming.com/johnkoza.html) and then move on through the ever-expanding body of research that grew from there. Or if you *really* get serious about this stuff, you will probably want to start all the way back with [**John Holland**](http://en.wikipedia.org/wiki/John_Henry_Holland), who as many of you may already know, is generally considered the "father" of research into the field of **Genetic Algorithms**.   
  
**ECJ** targets **Java**, because large scale evolutionary processes often need to interoperate seamlessly between workstations, which may or may not be hosting Windows, and back-end "heavy metal" hosting one of the **UNIX/LINUX** variants. Such environments are typical in many large enterprise and academic research settings, as you all well know (or know all too well!). If that kind of heterogeneous environment describes where *YOU* develop complex systems to solve difficult problems, then you will definitely want to trot on over to Sean's lab and familiarize yourself with the **Java** "source" code.   
  
On the other hand..., if you develop your software for homogenous environments with **Windows7 and Server 2003/2008/2008R2**, then you may want to consider downloading the converted **Open Source** **C#** code hosted here to take full advantage of tight integration with the latest features provided by **.NET 4.0**.   
  
Obviously I won't be able to deliver anywhere near the same level of support that a dedicated team can provide (almost none at all in fact). But if you are willing to work hard at it, or if you just want to experiment with the basics, everything you need to go the full distance is (or will shortly be) available in the download. I didn't leave anything out. But I should repeat the warning given above: some things have *NOT YET* been covered in the unit tests. I haven't had the time, for example, to test whether the Master / Slave communications are working (it uses simple sockets, so I'm not too worried about that). Same goes for Island Exchange for speciation, coevolution, and maybe another thing or two. The **ECJ** team was actively revamping some of the code for a robotics exercise while I was still in the middle of the conversion. Those gaps in coverage will be closed in due course.   
  
**I'll also revisit the ECJ SVN repository in coming weeks to pick up any new bits that show up and integrate any bug fixes and significant changes.**   
  
The goal for this project is to try and deliver a conversion of the latest bits as they accrue in **ECJ**, while maintaining a robust and flexible platform on which to develop in **C#, F#, IronPython, IronRuby, VB.NET** and so on. I will also be keeping my eye out for other juicy plums that will integrate nicely with this software. In fact, **ECJ** has a sibling project for **"Multi-Agent Simulation of Neighborhoods, or Networks (or something)" (aka MASON)** that fits the bill perfectly. And integration with [**Microsoft's Project Trident**](http://research.microsoft.com/en-us/collaboration/tools/trident.aspx) collaborative scientific research platform seems like a *no-brainer*. But there are also many other interesting possibilities. **Stay tuned!**

**NOTE: This software is a central feature in a much larger development picture that is partially laid out in the Vision statement.**

**BTW: BraneCloud** is linked to a registered domain (net|com) for future things like **name/schema** resolution in the **cloud**. So the only thing I ask if you want to derive your own fork from this one is that you **please use a different root namespace** to avoid collisions. Other that that, you are free to do with this what you will, so long as it abides by the stated licensing terms:

* **ECJ License (AFL 3.0)**
* **BraneCloud.Evolution.EC (Apache 2.0)**

*(Sean informs me that he may eventually be switching to the Apache license when he gets around to it.)*