



Mono State of the Union

FOSDEM 2012

Miguel de Icaza

(miguel@xamarin.com)

Agenda



- Mono in 2011
- Roadmap and Mono 2.12
- Help Wanted

Mono Team in 2011

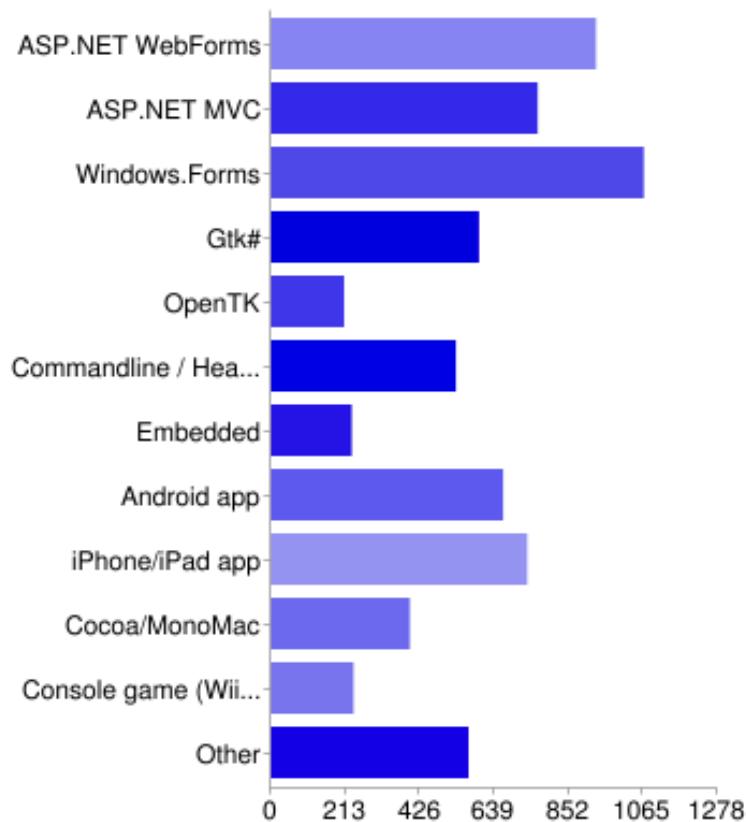


- Attachmate acquires Novell
- Mono team laid off (May)
- Xamarin Launched
 - Maintainers of Mono
 - Got license to all Mono IP
 - Xamarin sells proprietary software

Mono Download Survey



What type of application will you run on Mono?



ASP.NET WebForms	930	24%
ASP.NET MVC	763	19%
Windows.Forms	1067	27%
Gtk#	595	15%
OpenTK	209	5%
Commandline / Headless	529	13%
Embedded	231	6%
Android app	664	17%
iPhone/iPad app	733	19%
Cocoa/MonoMac	397	10%
Console game (Wii, Xbox 360, etc)	236	6%
Other	565	14%

People may select more than one checkbox, so percentages may add up to more than 100%.

Mono on the Server



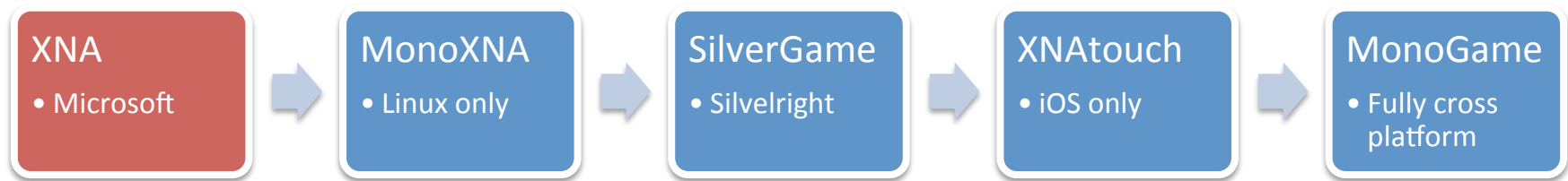
- Community maintained
- Commercial support from Sinenomine.Net
- Phalanger 3.0: Faster PHP running on Mono
 - Runs Wordpress, MediaWiki
 - On Linux, with Mono.
- Sgen GC
 - Massive performance/memory improvements

Mono in Gaming



- Unity3D:
 - 100 million plugin installs
 - 10%-20% of top sellers in Apple AppStore
- Delta Engine
- Sony PlayStation Suite
- Google Native Client
- Microsoft Kinectimals

MonoGame



Mono on the Desktop



- Gtk# continues to be our core toolkit
- Adding MonoMac – Native apps in OSX
- New XWT toolkit
- FUD wars

Mono on Mobile



- Proprietary offerings
 - Thousands of applications published
- Growing developer base
 - 100/day September 2011
 - 200/day August 2011
 - 300/day November 2011
 - 400/day January 2012



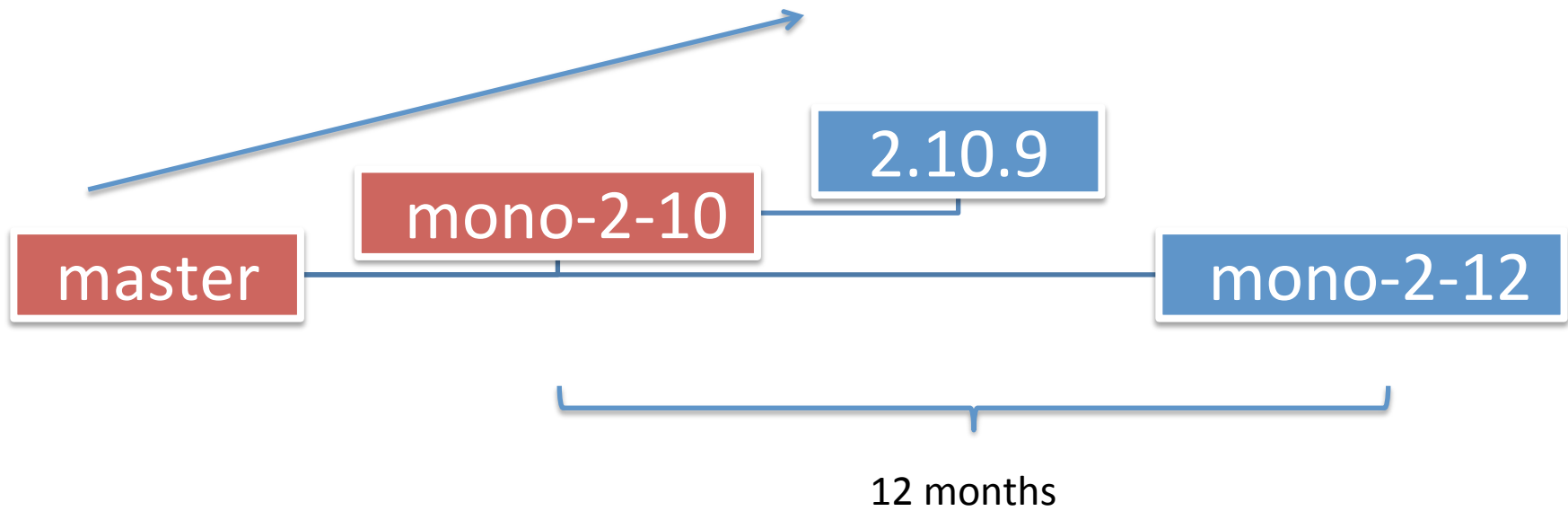
ROADMAP

Reducing Mono's Inventory



- Currently:
 - Stable branches fork early on
 - Maintained long-term
 - Continuous bug fixes backported
- **master** has a big inventory of features
 - Little testing
 - Long QA periods

Currently



Next



- Features Developed in Branches
 - Landing those changes when they stabilize
- Continuous Integration:
 - Expose our Wrench Framework to the world
 - Package Branches for use in Linux, Mac, Window
- Packaging:
 - Work closer with packagers
 - Offer official openSUSE/Debian/Fedora/Ubuntu on release date

Upcoming Mono Releases



- Mono 2.11:
 - Preview Release for Mono 2.12
 - Will keep in testing for 2-3 months
 - Work with upstream maintainers
- Mono 2.12:
 - Stable release
 - master always stable

MonoDevelop



- Powerful IDE with many features
 - Used extensively
- 1. Focus on stability and bug fixing
- 2. New auto-complete engine
 - Shared effort with SharpDevelop
- 3. April: start work on new UI face lift
 - Working with professional designers on a new UI



MONO 2.12 – IN GENERAL

Some Features



- Silverlight's System.Json everywhere
- Portable Class Library profile
 - Allows sharing same library across platforms
- .NET 4.0 Static Code Contracts Analyzer
- TPL DataFlow Preview
- MacOS X 64 bit support
- WinRT APIs



MONO 2.12 – C# 5.0

C# 5.0 and Async Programming

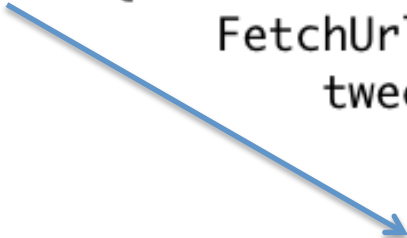


- Mono **master** has a complete C# 5 Compiler
- Turns repetitive callback-based async programming into linear programming
 - The compiler rewrites the code into a state machine
 - Tasks are scheduled on the main thread
 - But execution can be customized

Current Trends in Async Programming



```
void DownloadTweets ()
{
    FetchUrl (tweetUrl, result => {
        tweetDb.Populate (ParseJsonResults (result), () => {
            View.BeginInvokeOnMainThread (delegate {
                View.ReloadData ();
            });
        });
    });
}
```

A blue arrow originates from the left side of the slide and points to the `View.ReloadData ();` line within the nested lambda expressions of the `FetchUrl` method call.

The C# 5.0 version



```
async void DownloadTweets ()
{
    var result = await FetchUrl (tweetUrl);
    await tweetDb.Populate (ParseJsonResults (result));
    View.ReloadData ();
}
```

```

void DownloadTweets ()
{
    FetchUrl (tweetUrl, (result) => {
        if (result == null)
            View.InvokeOnMainThread (delegate {
                ShowError ("Could not download tweets");
            });

        tweetDb.Populate (ParseJsonResults (result), (error) => {
            if (error){
                Tweet.UpdateLastRead (lastValidCode, errorPost => {
                    if (errorPost)
                        View.InvokeOnMainThread (delegate {
                            ShowError ("Twitter is down");
                        });
                });
            } else {
                View.BeginInvokeOnMainThread (delegate {
                    View.ReloadData ();
                    lastValidCode = currentCode;
                });
            }
        });
    });
}

```



Error Handling



- Too much noise
- Easy to make mistakes
- Propagating errors becomes cumbersome
 - Tracking multiple levels of state
- Humans are doing what tools should be doing
- C# 5 moves the burden to the compiler



```
async void DownloadTweets ()
{
    var tweets = await FetchUrl (tweetUrl);
    if (tweets == null){
        ShowError ("Could not download tweets");
        return;
    }
    if (!await tweetDb.Populate (ParseJsonResults (result))){
        if (!await Tweet.UpdateLastRead (lastValidCode))
            ShowError ("Twitter is down");
    } else {
        View.ReloadData ();
        lastValidCode = currentCode;
    }
}
```


Async



- Natural fit for Desktop and Mobile app
 - They need to be responsive to user input
 - If API takes > 50 ms, make the API async
- Async becoming popular for Servers too
 - Node.js popularized this (no threads in JS)
 - Blend callbacks + threads as needed

Build on Task Parallel Library



- Every async method returns void or Task<T>

```
async Task<Stream> FetchUrl (string url)
{
    var uri = new Uri (url);
    var httpRequest = CreateRequest (uri);
    return await httpRequest.GetResponse ();
}
```



MONO 2.12 – .NET 4.5

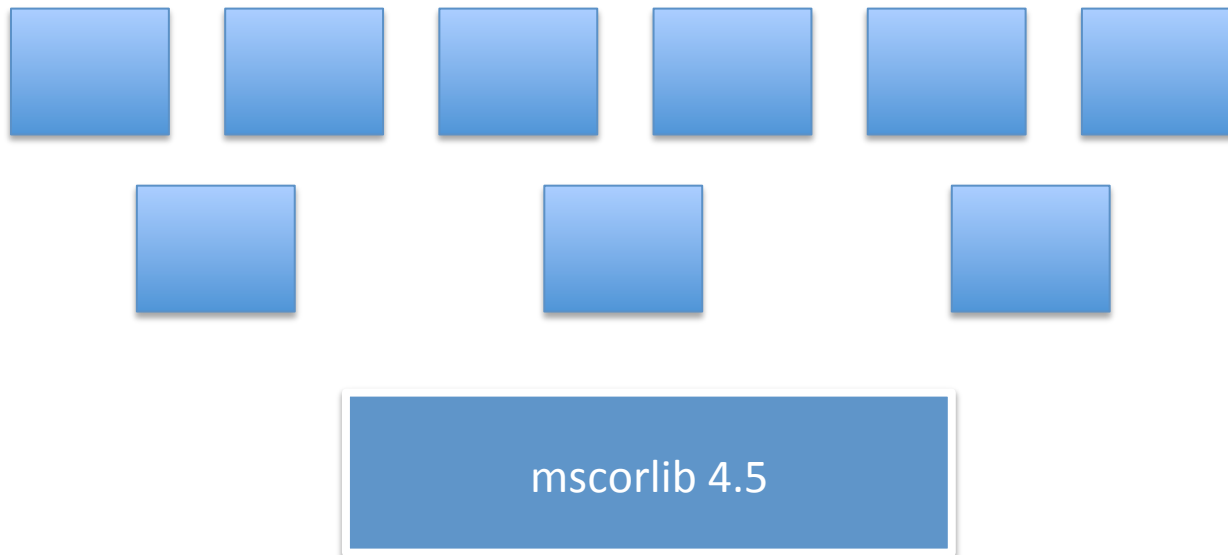
.NET 4.5



- Strict superset of .NET 4.0
 - Assemblies are versioned with 4.0 strong names
- New APIs for Async programming
 - System.IO
 - System.Net, System.Net.Sockets
- New Http Stack: System.Net.Http
 - Async-ready from the start
 - Nicer, simpler API

Debian's Dream – WinRT changes

- Clean up:
 - Dependencies
 - Obsolete code
- Evolutionary step
 - Same foundation today
 - Type forwarders



Ongoing work with ECMA to specify the new class libraries.



MONO 2.12 – SGEN

SGen – New Garbage Collector



- Generational collector
- Servers and HPC loads:
 - 10-50% performance boost
 - 10-30% memory reduction
- Foundation for all new Mobile Developments
 - Easier to tune to our needs

Using SGen



- Use:
 - `mono -sngen`
- Default in Mono 3.0

MonoDevelop Directions



- Fixing Intellisense
- Compiler as a Service
- Nrefactory
- Joint work with the SharpDevelop team