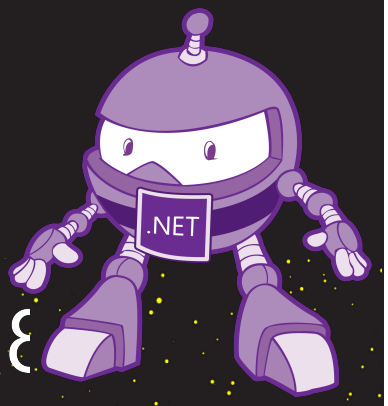

.NET Conf 2018
Discover the world of .NET
dotnet new
.NET

Retro.NET

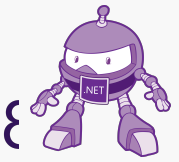
Miguel de Icaza – miguel@microsoft.com
@migueldeicaza



.NET Conf 2018

Building Great Terminal Applications

- Integrate with your environment
- Provide a good user experience
  - Command Line
  - Text User Interfaces
- Distribute your work

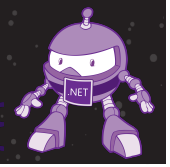


.NET Conf 2018

Command Line Parsing

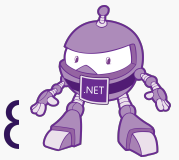
.NET

.NET Conf 2018



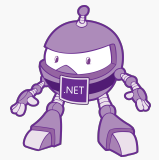
Mono.Options

- Command-line arguments parsing made easy
  - Handle option flags
  - Options with arguments
  - Nested commands
- Support for common idioms
  - Use of "--" for long options
  - Use of "-" for short options
  - `command --path=/usr/local`
- NuGet: Mono.Options
  - Implemented as a single C# file



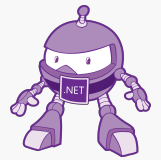
Mono.Options - Basic Usage

```
bool ipv4, ipv6;
bool wantListener;   // Create a listener
bool verbose;        // verbose
bool bonjour;
bool useTls, useUdp;
string localPort, localAddr, psk;

OptionSet options =
    new OptionSet () {
        { "4" , "Use IPV4", v => ipv4 = true },
        { "6" , "Use IPV6", v => ipv6 = true },
        { "b", "Use Bonjour", v => bonjour = true },
        { "l|listener", "Create a listener to accept inbound connections",
          v => wantListener = true },
        { "p=|port=", "Use a local port for outbound connections",
          v => localPort = v },
        { "s=|localaddr=", "Sets the local address for outbound connections",
          v => localAddr = v },
        { "t|tls", "Add TLS/DTLS as applicable", v => useTls = true },
        { "u|udp", "Use UDP instead of TCP", v => useUdp = true },
        { "v|verbose", "Verbose", v => verbose = true },
        { "k=|psk=", "Specify the TLS Pre-Shared Key", v => psk = v },
        { "h|help", "Show this help", v => ShowHelp (0, true) },
};
```



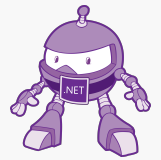
Some Command Invocations

```
$ ncsharp -6 -p 3000 --psk=secret -t www.microsoft.com https

$ ncsharp --port=3000 --tls www.microsoft.com https
```

Mono.Options – Basic Invocation

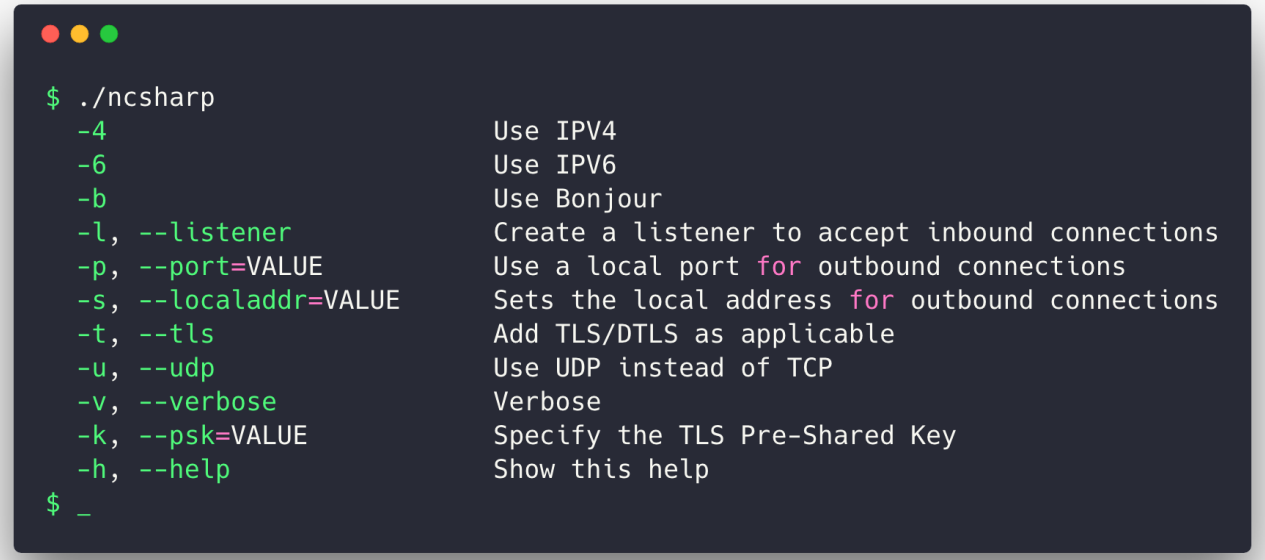
```
var rest = options.Parse (args);
string hostname, port;
switch (rest.Count){
    case 1:
        hostname = rest [0];
        port = "http"; break;
    case 2:
        hostname = rest [1];
        port = rest [2];
    default:
        ShowHelp ();
        break;
}
```



Mono.Options – Show help

- Call WriteOptionDescriptions:

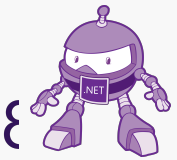
```
options.WriteOptionDescriptions(Console.Error);
```



Nested Commands – Command Suites

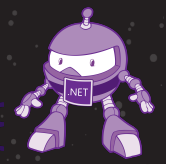
```
var git = new CommandSet ("git") {
    "usage: git [--version] ... <command> [<args>]",
    "",
    "Common Options:",
    { "version",
        "show version info",
        v => showVersion = v != null },
    { "help",
        "show this message and exit",
        v => showHelp = v != null },
    "",
    "These are common Git commands used in various situations:",
    "",
    "start a working area (see also: git help tutorial)",
    new Command ("clone",
                 "Clone a repository into a new directory") {
                 Run = v => Clone ()
    },
    new Command ("init",
                 "Create an empty Git repository or reinitialize an existing one") {
                 Run = v => Init ()
    },
    new CommandSet ("submodule") {
        "Submodule core command",
        new Command ("init", "Initializes the submodule stuff"),
        "Additional commands",
        new Command ("add", "use -b branch to specify things"),
        new Command ("remove", "Removes a submodule"),
    }
};
```

```
mono demo.exe
usage: git [--version] ... <command> [<args>]

Common Options:
      --version             show version info
      --help                show this message and exit

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
        clone               Clone a repository into a new directory
        init                Create an empty Git repository or reinitialize
                            an existing one
Submodule core command
        submodule init      Initializes the submodule stuff
Additional commands
        submodule add       use -b branch to specify things
        submodule remove    Removes a submodule
```



Terminal User Interaction

.NET

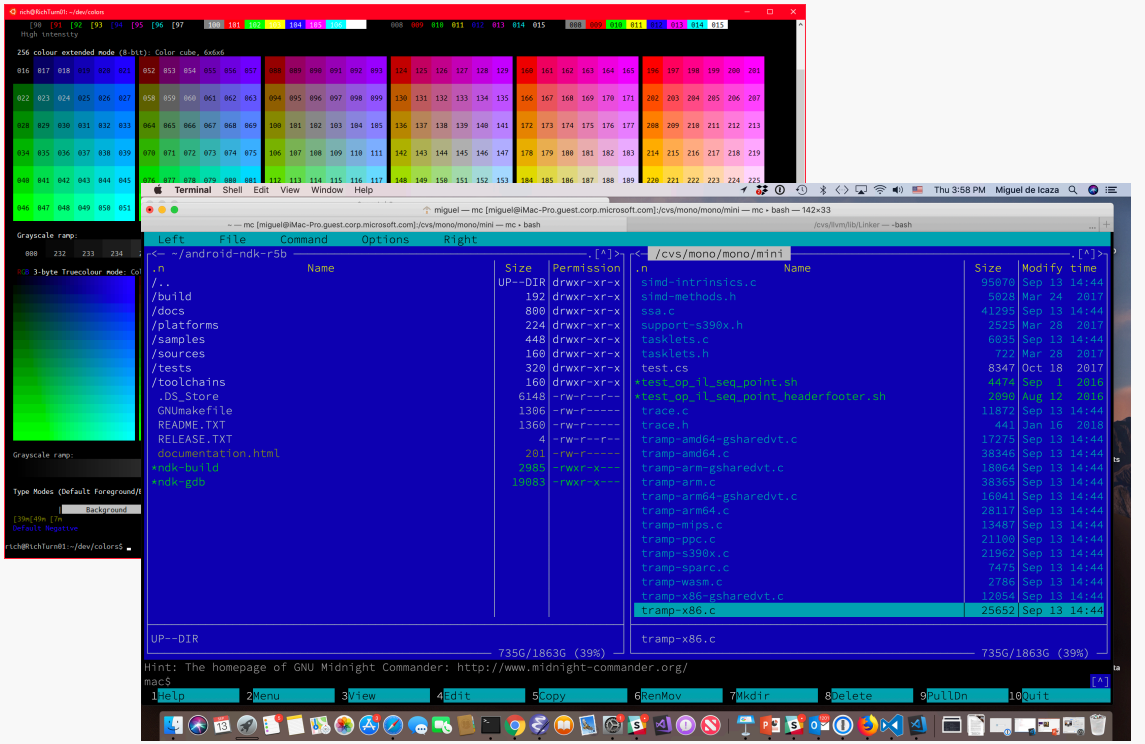


.NET Conf 2018

From VT100 to Terminal.app to WinConsole

VT100 Terminal

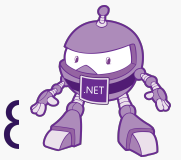
By Jason Scott - Flickr: IMG_9976, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=29457452



.NET Conf 2018

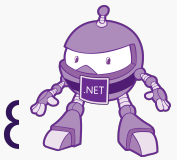
Terminal Styles

- Windows
  - Windows Console supports colors
- VT-100 Family of terminals
  - Original VT-100 (dialup modems, some SSH clients)
- Xterm-based
  - Based on the original VT-100 extended over the years
  - Mouse event support
  - Some contain colors
  - macOS default terminal, most modern Linux terminals
- Other Unix-supported terminals



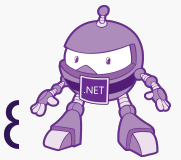
How Unix Detects a Terminal

- TERM environment variable
- Terminfo is a system wide database
  - Lists terminal capabilities
  - Escape sequences used to render information
  - Describes character sequences produced by terminals (cursor keys, function keys, etc).
- Dimensions
  - `LINES`, `COLS` variables
  - `terminfo` capabilities
  - `ioctl` operation on the standard input



Colors and Attributes

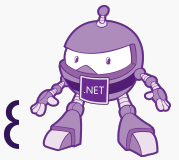
- Black and white
  - Blink, bold, reverse operations
- 8-colors (some configurations and ncurses)
- 16-colors (foreground, background)
- 256-color palette (216 colors+16 ANSI+24 gray)
- 24bit true color (16 million colors)
  - Supported in terminal emulators, not described in most databases



Unix Console Libraries

- ncurses – an evolution of curses library
  - Actively maintained
  - Mouse support
  - Often out of date across distributions
  - ABI is determined at compile time

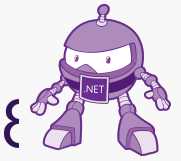
- Raw: access terminfo database directly



System.Console

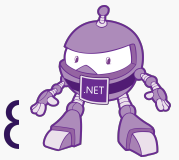
- Basic console input and output
  - Write, WriteLine
  - Read, ReadLine
  - Interactive: ReadKey
- Supports 16 colors
- Cursor positioning

- Supported on Windows, Linux, Mac
- On .NET Framework, .NET Core and Mono



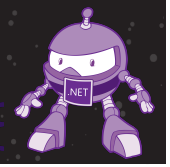
Console – very limited

- API is limited
  - Not very powerful
  - Not a good match to Unix idioms
  - Not even a good match on Windows (limited event handling, screen display limitations, no mouse support)
- Suitable for coloring output (red for errors, etc)
- No framework for reasoning at higher levels



Interactive Line Editing

.NET

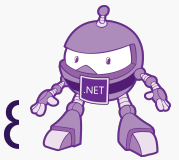


.NET Conf 2018

Interactive Line Editing

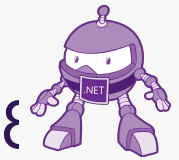
- A user-friendly replacement for Console.ReadLine.
- Editing with the cursor keys
- Emacs key bindings (copy/paste/transpose, etc)
- Saved history
- Search in history

- Optional Completion
- Built on top of `System.Console`



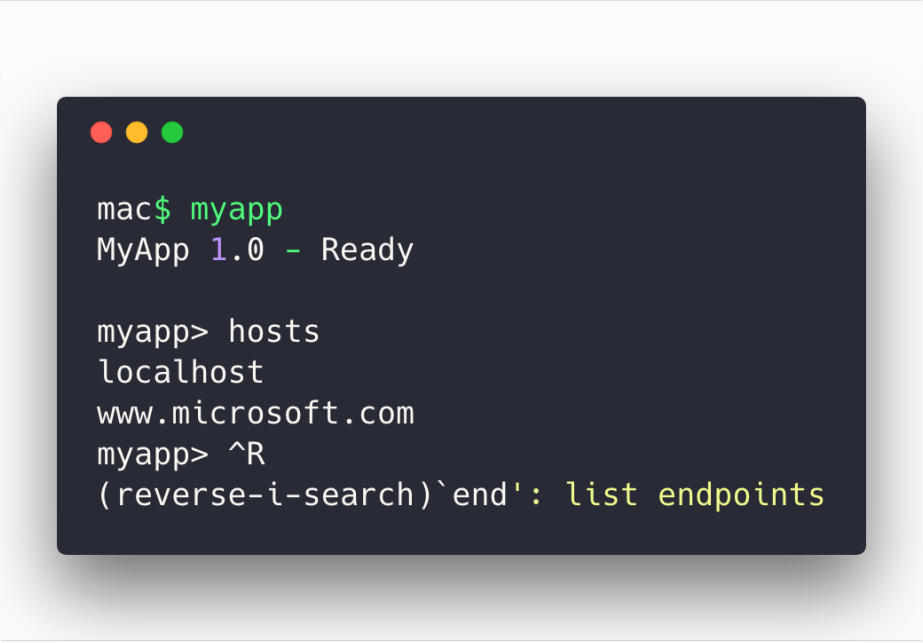
Basic Usage

```
var prompt = "myapp> ";
var editor = new Mono.Terminal.LineEditor ("myappHistory", histsize: 300);
editor.Edit (prompt, "");
```

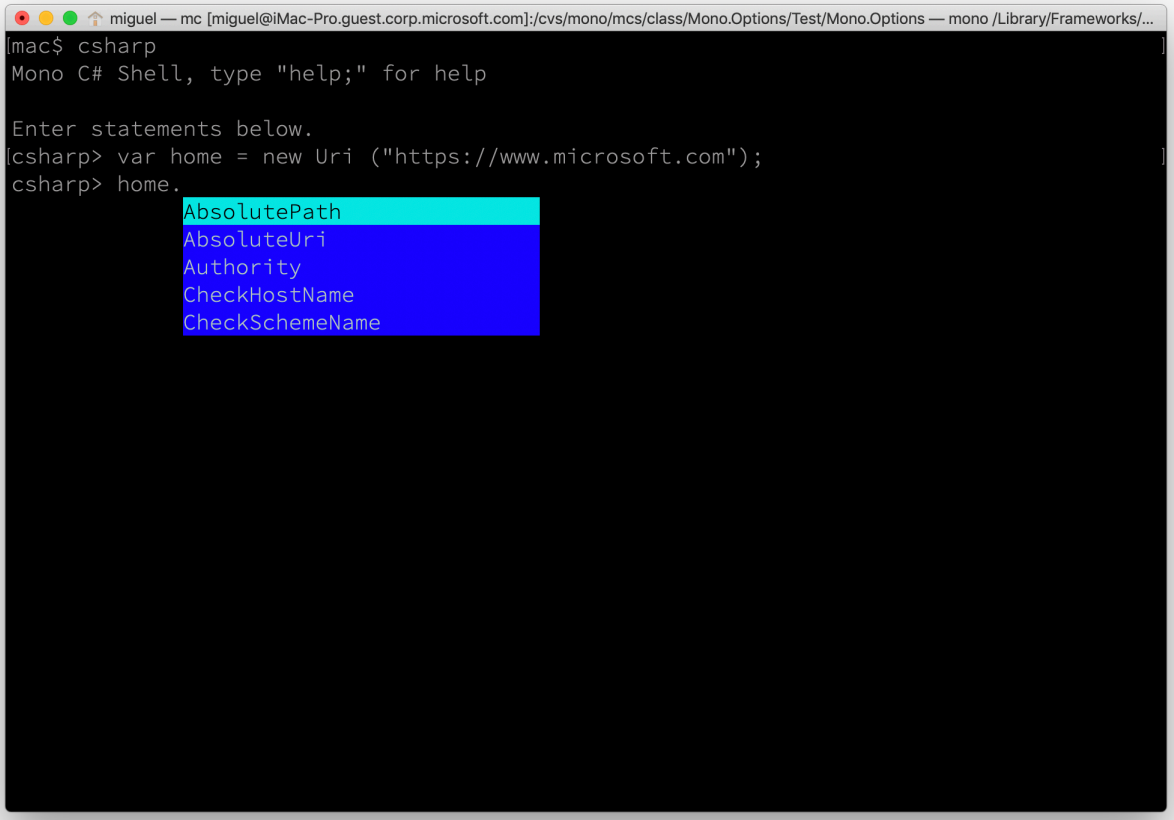



History and Incremental Search

- Access history with the arrow keys (up/down)
- Reverse incremental search (C-r)

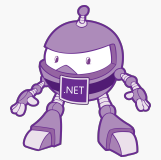


Completion


miguel — mc [miguel@iMac-Pro.guest.corp.microsoft.com]:/cvs/mono/mcs/class/Mono.Options/Test/Mono.Options — mono /Library/Frameworks/...
[mac$ csharp
Mono C# Shell, type "help;" for help
Enter statements below.
[csharp> var home = new Uri ("https://www.microsoft.com");
csharp> home.
AbsolutePath
AbsoluteUri
Authority
CheckHostName
CheckSchemeName

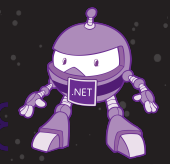
Adding Completion

```
editor = new Mono.Terminal.LineEditor ("csharp", 300) {
    HeuristicsMode = "csharp"
};

editor.AutoCompleteEvent += delegate (string currentInput, int pos){
    string prefix = null;

    string complete = currentInput.Substring (0, pos);

    string [] completions = evaluator.GetCompletions (complete, out prefix);

    return new Mono.Terminal.LineEditor.Completion (prefix, completions);
};
```



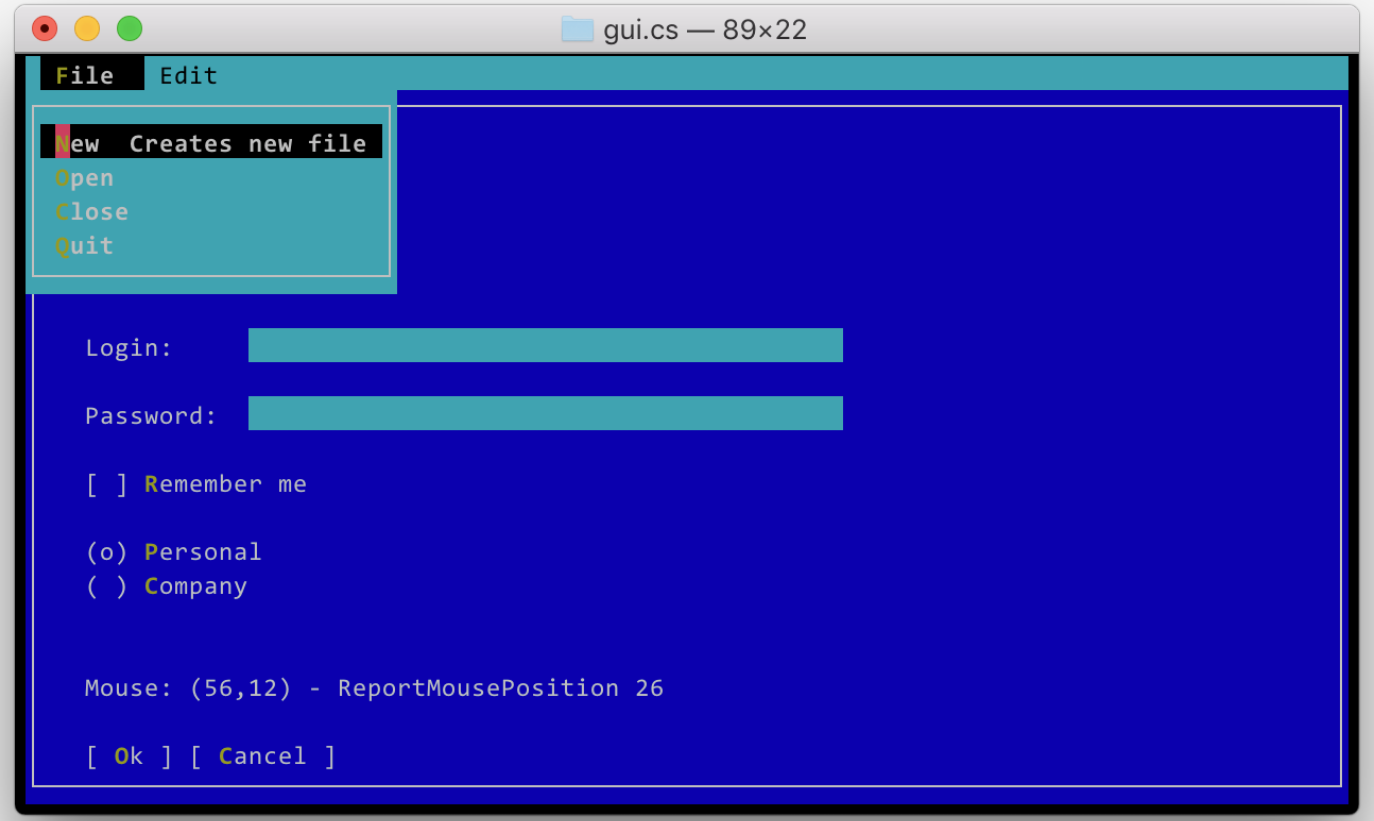
Text User Interfaces

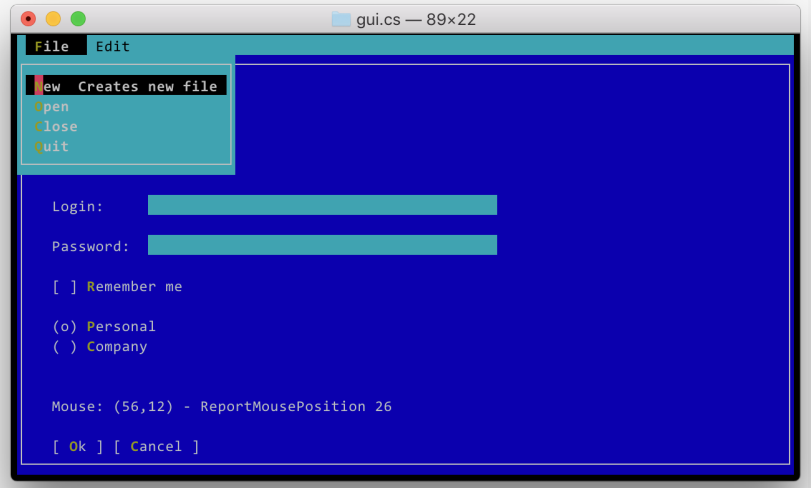
Terminal.Gui (gui.cs) .NET



.NET Conf 2018

Gui.cs in action

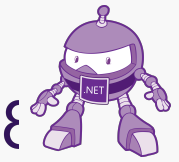





```cs
using Terminal.Gui;

class Demo {
    static void Main ()
    {
        Application.Init ();
        var top = Application.Top;

        var win = new Window (new Rect (0, 1, top.Frame.Width, top.Frame.Height-1), "MyApp");
        top.Add (win);

        var menu = new MenuBar (new MenuBarItem [] {
            new MenuBarItem ("_File", new MenuItem [] {
                new MenuItem ("_New", "Creates new file", NewFile),
                new MenuItem ("_Close", "", () => Close ()),
                new MenuItem ("_Quit", "", () => { if (Quit ()) top.Running = false; })
            }),
            new MenuBarItem ("_Edit", new MenuItem [] {
                new MenuItem ("_Copy", "", null),
                new MenuItem ("C_ut", "", null),
                new MenuItem ("_Paste", "", null)
            })
        });
        top.Add (menu);

        win.Add (
            new Label (3, 2, "Login: "),
            new TextField (14, 2, 40, ""),
            new Label (3, 4, "Password: "),
            new TextField (14, 4, 40, "") { Secret = true },
            new CheckBox (3, 6, "Remember me"),
            new RadioGroup (3, 8, new [] { "_Personal", "_Company" }),
            new Button (3, 14, "Ok"),
            new Button (10, 14, "Cancel"),
            new Label (3, 18, "Press ESC and 9 to activate the menubar"));

        Application.Run ();
    }
}
```

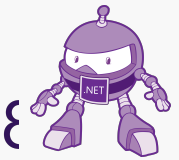
Gui.cs Basics

- Toolkit for building Text User Interfaces
- Build full-screen text applications
- Blending the past and the present
  - Handle limitations of old terminals or emulators (lack of Alt/Meta-key), colors
  - Supports modern features (terminal resize, events, application main loop, async)
- Common UI Controls
- Borrows ideas from:
  - iOS's UIKit design
  - Various other UI toolkits (Gtk+, WinForms)



Application.Run – the Application Loop

- Application.Run provides a main loop
  - Process input events (keyboard, mouse)
  - Timers
  - Idle callbacks
  - On Unix can monitor file descriptor readiness availability (read, write, notification)
  - Responds to window-size changes, Unix suspend (control-z)
- Routes events to the UI elements
- Handles Focus (via keyboard or mouse)
- Redraws affected areas of the screen

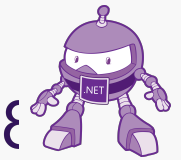


Keyboard Input

- On Windows everything keyboard works as expected
- Some Unix Terminals and remote Systems
  - Sometimes Function keys not mapped, or not obvious how to map
  - Alt-letter not configured to be sent

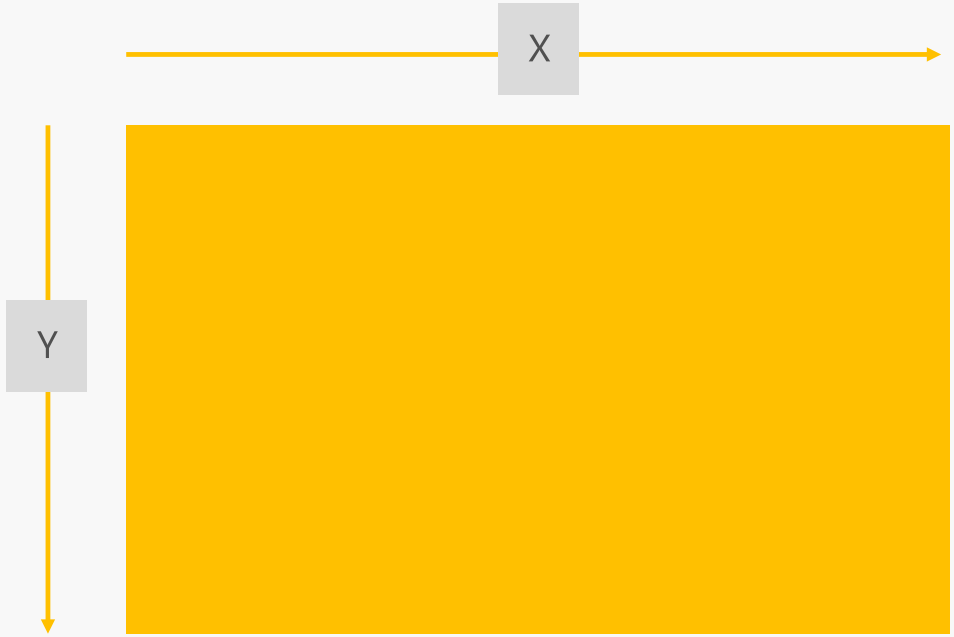
- Gui.cs Mappings
  - ESC + number is mapped to FunctionKey-numer
  - ESC + letter is mapped to Alt-letter

- Many “hot” keys are accessed with Alt-letter

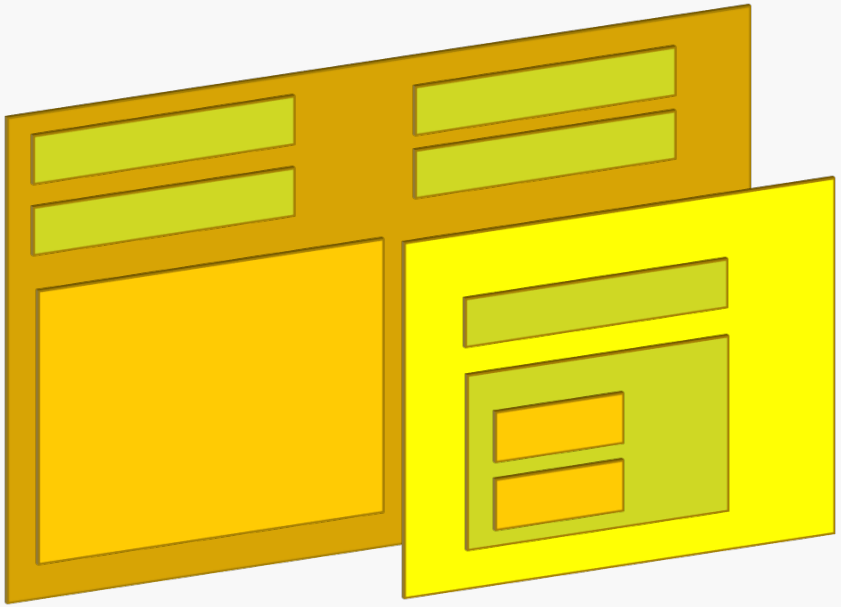


Views – at the Core

- Visual Elements that have a Frame
  - Assigned via X, Y, Width, Height properties, or setting an absolute frame
- The composition unit for user interfaces

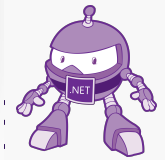


Views – can have nested Views



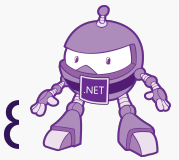
Core Layout Types

```
public struct Rect {
    public Rect (int x, int y, int width, int height);
    public int X, Y, Width, Height;
}

public struct Point {
    public Point (int x, int y);
    public int X, Y;
}

public struct Size {
    public Size (int width, int height);
    public int Width, Height;
}
```



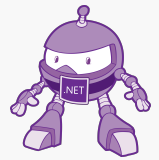
Layout

- Absolute positioning, create Views with Rect()
  - You must re-layout your views manually on LayoutSubview method.
- Smart positioning
  - Set the X, Y properties (they are of type “Pos”)
  - Set the Width, Height properties (they are of type “Dim”)
  - Integers are implicitly convertible to them



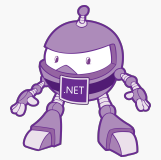
```
public class Pos {
  static Pos Percent (float n);
  static Pos AnchorEnd (int margin = 0);
  static Pos Center ();
  static implicit operator Pos (int n);
  static Pos operator + (Pos left, Pos right);
  static Pos operator - (Pos left, Pos right);

  static Pos Left (View view);
  static Pos Top (View view);
  static Pos Right (View view);
  static Pos Bottom (View view);

  static Pos X (View view);
  static Pos Y (View view);
}
```

```
public class Dim {
  static Dim Percent (float n);
  static Dim Fill (int margin = 0);

  static implicit operator Dim (int n);
  static Dim operator + (Dim left, Dim right);
  static Dim operator - (Dim left, Dim right);

  static Dim Width (View view) => new DimView (view, 1);
  static Dim Height (View view) => new DimView (view, 0);
}
```

Allows for nice layout rules in code and anchors

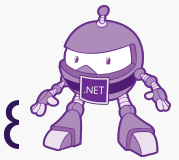
```
var login = new Label ("Login: ") {
  X = Pos.Center (),
  Y = 6
};
var password = new Label ("Password: ") {
    X = Pos.Left (login),
    Y = Pos.Bottom (login) + 1
};
var loginText = new TextField ("") {
    X = Pos.Right (password),
    Y = Pos.Top (login),
    Width = 40
};
var passText = new TextField ("") {
    Secret = true,
    X = Pos.Left (loginText),
    Y = Pos.Top (password),
    Width = Dim.Width (loginText)
};

dirEntry = new TextField ("") {
    X = Pos.Right (passText),
    Y = Pos.Bottm (passText) + 2
    Width = Dim.Fill () - 1
};
```



Toplevel Views

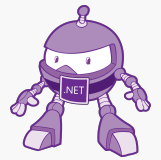
- Special subclass of View
- Participate in modal event handling
- In particular, the parameter to Application.Run()

- Toplevel – basic toplevel
- Window – toplevel with a border and title



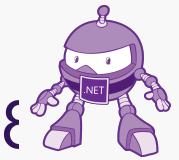
Focus and events

```
// Focus properties
public virtual bool HasFocus { get; }
public virtual bool CanFocus { get; set; }

// Input events
public virtual bool ProcessHotKey (KeyEvent kb);
public virtual bool ProcessKey (KeyEvent keyEvent);
public virtual bool ProcessColdKey (KeyEvent keyEvent);
public virtual bool MouseEvent (MouseEvent mouseEvent);
```

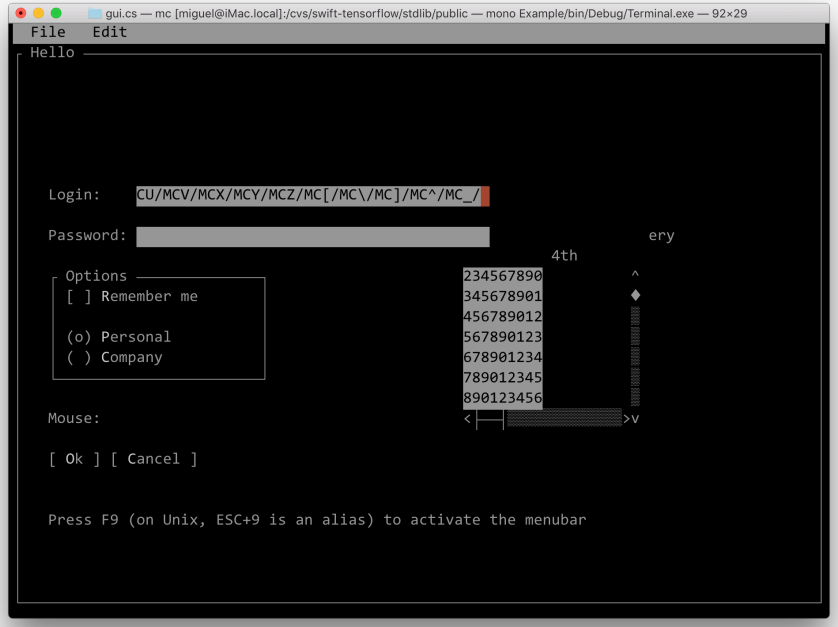


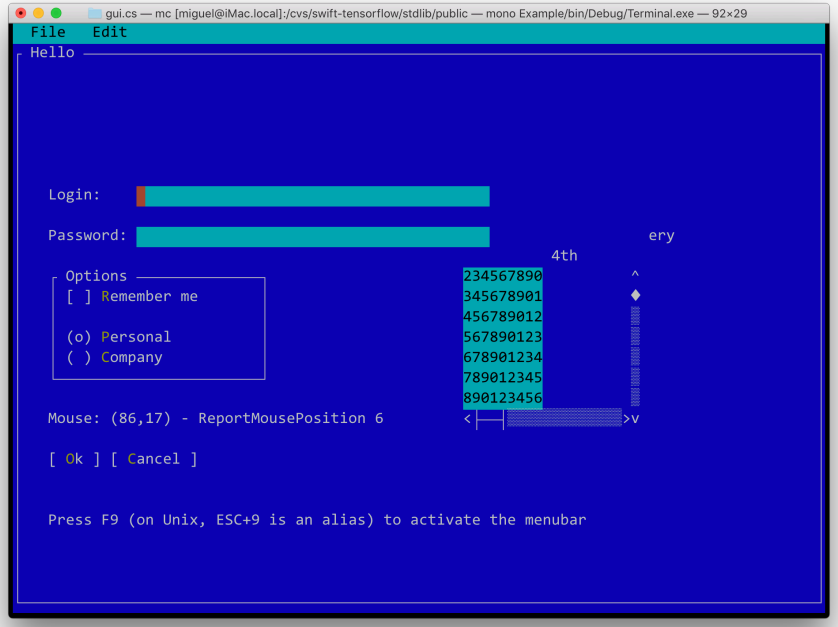
Colors

- Attribute class represents full cell color
  - Background + Foreground for color, or basic black and white description
  - Wraps an Int32
  - Created on demand based on the colors used
- These are opaque constants
- Some are created at startup for styling, but can be created on demand and used



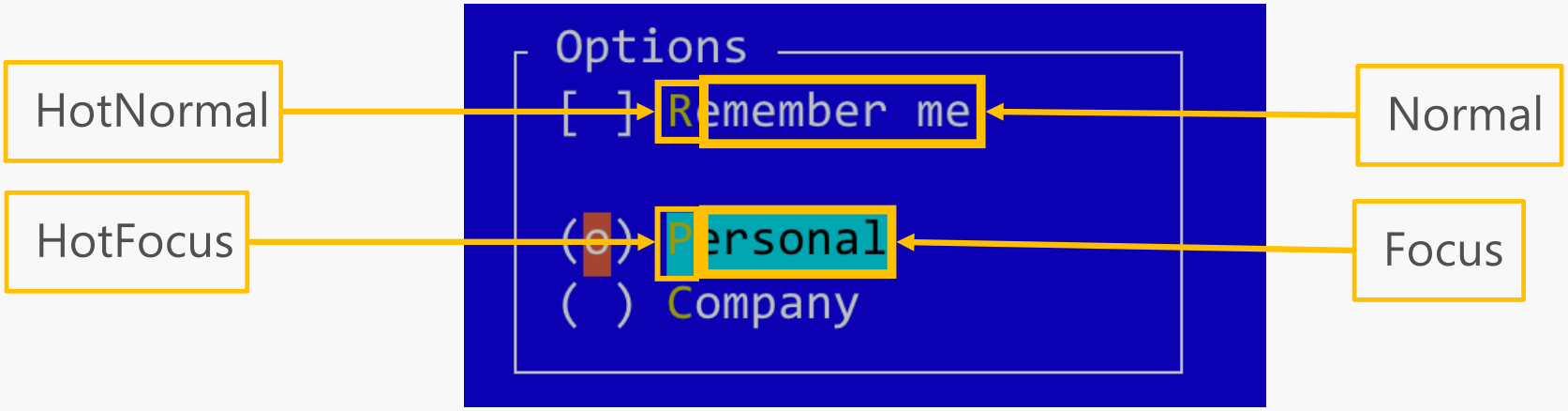
VT100 vs Xterm





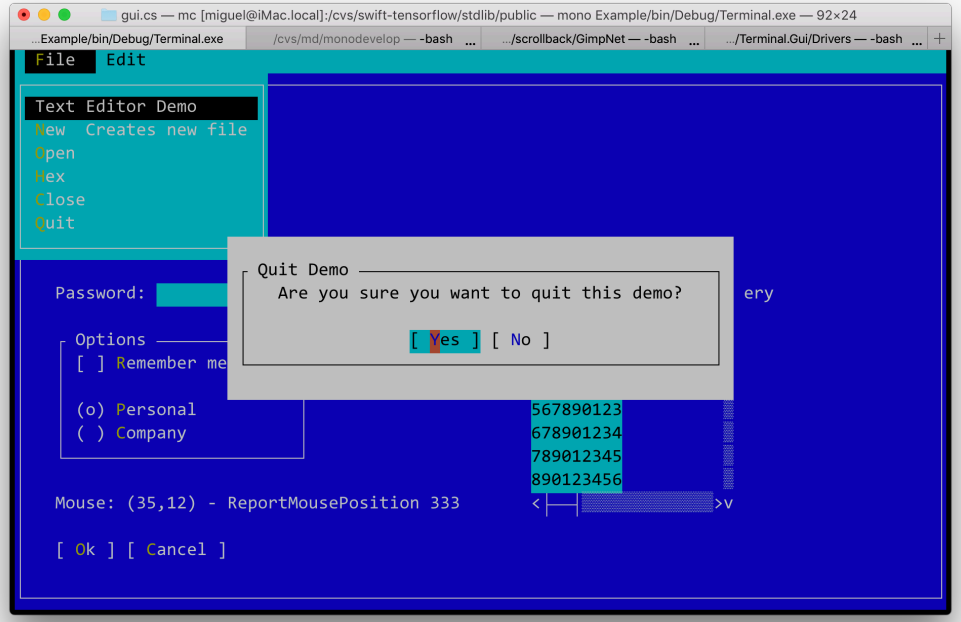
ColorScheme

Most Views need various attributes to render



Built-in Styles

```
/// <summary>
/// The default ColorSchemes for the application.
/// </summary>
public static class Colors {
    /// <summary>
    /// The base color scheme, for the default toplevel views.
    /// </summary>
    public static ColorScheme Base;

    /// <summary>
    /// The dialog color scheme, for standard popup dialog boxes
    /// </summary>
    public static ColorScheme Dialog;

    /// <summary>
    /// The menu bar color
    /// </summary>
    public static ColorScheme Menu;

    /// <summary>
    /// The color scheme for showing errors.
    /// </summary>
    public static ColorScheme Error;

}
```



.NET Conf 2018

Common Views

Label

Button
Clickable button, provides a “hotkey”

Entry

CheckBox

ProgressBar

RadioGroup

TextField
single-line text editing

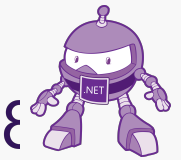
TextView
multi-line text editing or viewing

ScrollView
Virtualizes views added to it

Menu, MenuBar
Toplevel menus

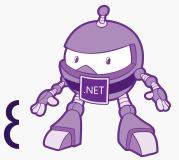
FrameView
container that draws a frame around its children

HexView
Hexadecimal dump + binary dump



Dialogs – Use For Modal Interactions

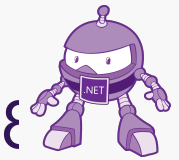
```
var dialog = new Dialog (
    "Title", 50, 20,
    new Button ("Ok", is_default: true) {
         Clicked = () => { Application.RequestStop (); }
    },
    new Button ("Cancel") {
         Clicked = () => { Application.RequestStop (); }
    });
Application.Run (d);
```



MessageBox

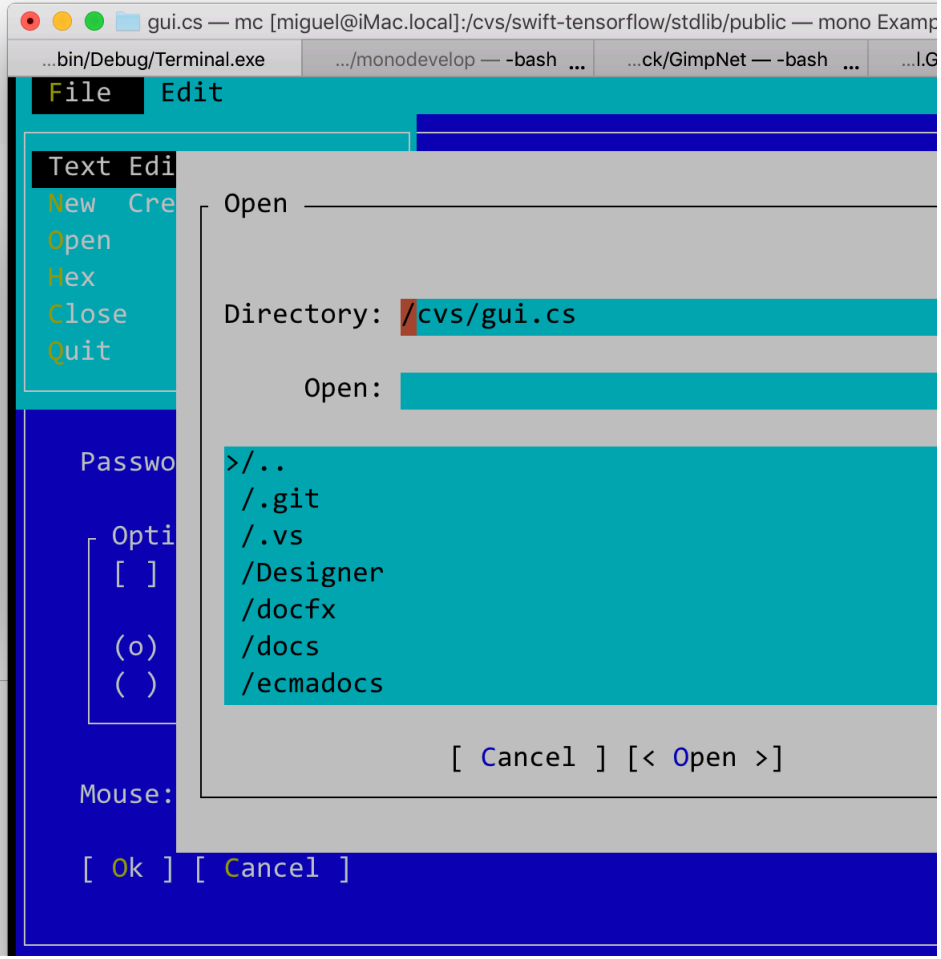
- Simple wrapper to ask questions

```
bool ShouldQuit ()
{
    var n = MessageBox.Query (50, 7, "Quit Demo",
        "Are you sure you want to quit this demo?",
        "Yes", "No");
    return n == 0;
}

void ShowError ()
{
    MessageBox.ErrorQuery (50, 5, "Error",
                           "File Not Found", "Ok");
}
```



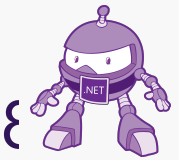
File Open and File Save Dialogs

```
var d = new OpenDialog ("Open", "Open a file") {
  AllowsMultipleSelection = true,
  CanChooseFiles = true,
  CanChooseDirectories = true
};

Application.Run (d);
foreach (var f in d.FilePaths)
   Console.WriteLine ("Selected " + f);
```



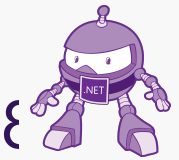
Async operations

- Gui.cs is thread aware, but not thread safe
  - Calls into Gui.cs APIs must be done in the same thread that Application.Run is running
- Async/Await
  - Comes with an async/await SyncContext
  - Just use async/await and Tasks to do background operations
- If you are not in an async context and must call
  - Application.MainLoop.Invoke is thread-safe
  - Use Application.MainLoop.Invoke (Action) to queue the Action to run on the UI thread



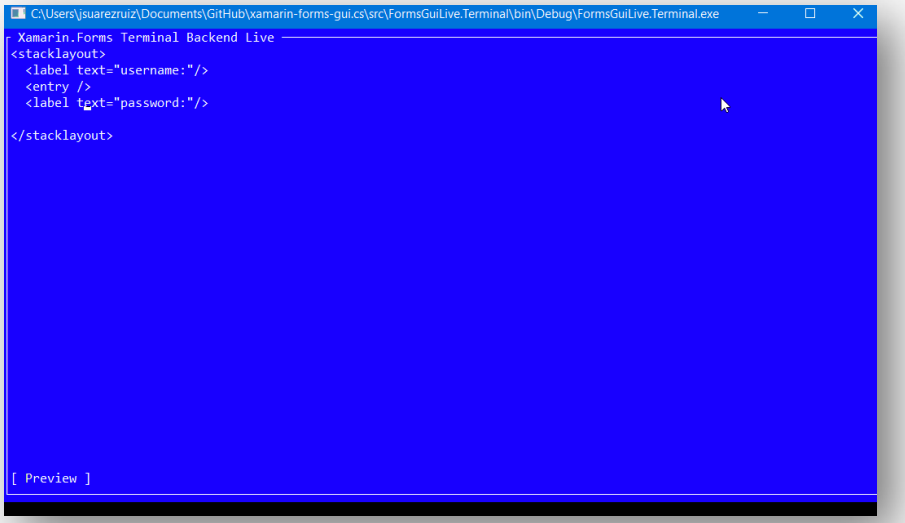
Drivers – abstracting the host console

- Curses Driver
  - Uses the “ncurses” library on Unix
- System.Console Driver
  - Works on Unix and Windows
  - Lack of Mouse support, some drawing limitations (corner)
- Native Windows Driver
  - P/Invokes directly into the Windows Console API – Complete
- Future for Unix
  - Avoid uses ncurses, implement a full driver based on terminfo capabilities

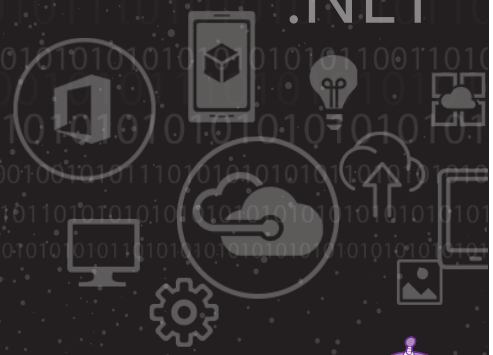


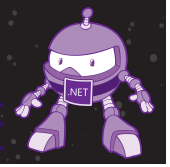
Javier Suarez built a Xamarin.Forms backend

https://github.com/jsuarezruiz/xamarin-forms-gui.cs



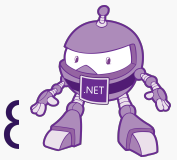
Creating Commands





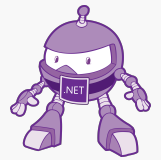
Turning your Console app into a Tool

```
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>netcoreapp2.1</TargetFramework>
  </PropertyGroup>


  <!-- Add to make a tool -->
  <PropertyGroup>
    <PackageId>MyPackageName</PackageId>
    <IsTool>true</IsTool>
    <PackAsTool>true</PackAsTool>
    <ToolCommandName>my-tool</ToolCommandName>
  </PropertyGroup>

</Project>
```

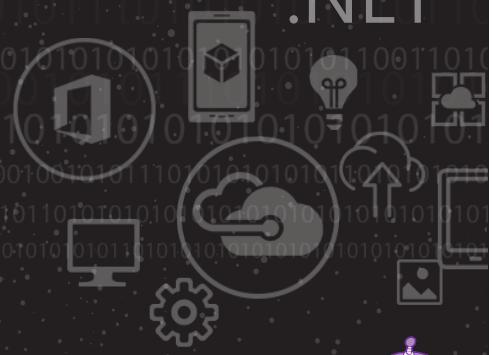


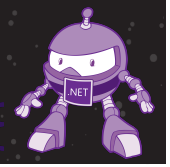
Pack and Install

```
$ dotnet pack
Microsoft (R) Build Engine version 15.8.166+gd4e8d81a88 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.

  Restoring packages for /private/tmp/bar/bar.csproj...
  Restore completed in 156.67 ms for /private/tmp/bar/bar.csproj.
  bar -> /private/tmp/bar/bin/Debug/netcoreapp2.1/bar.dll
  bar -> /private/tmp/bar/bin/Debug/netcoreapp2.1/bar.dll
  bar -> /private/tmp/bar/bin/Debug/netcoreapp2.1/publish/
  Successfully created package '/private/tmp/bar/bin/Debug/MyPackageName.1.0.0.nupkg'.

$ dotnet tool install --global --add-source bin/Debug MyPackageName
You can invoke the tool using the following command: my-tool
Tool mypackagename (version 1.0.0) was successfully installed
```

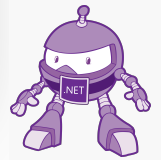
Single-contained executables





Mono can bundle single-file, self-contained executables

```
$ mkbundle --simple ncsharp.exe -o ncsharp
  -L /Library/Frameworks/Xamarin.Mac.framework/Versions/Current//lib/x86_64/full/
  --library /Library/Frameworks/Xamarin.Mac.framework/Versions/Current//lib/libxammac.dylib

Using runtime: /Library/Frameworks/Mono.framework/Versions/5.12.0/bin/mono
     Assembly: /cvs/NetCatNetwork/ncsharp.exe
     Assembly: /Library/Frameworks/Mono.framework/Versions/5.12.0/lib/mono/4.5/mscorlib.dll
     Assembly: /Library/Frameworks/Mono.framework/Versions/5.12.0/lib/mono/4.5/I18N.West.dll
     Assembly: /Library/Frameworks/Mono.framework/Versions/5.12.0/lib/mono/4.5/I18N.dll
     Library:  /Library/Frameworks/Xamarin.Mac.framework/Versions/Current//lib/libxammac.dylib
Generated ncsharp
$ ls -l ncsharp
-rwxr-xr-x  1 miguel  _lpoperator  9912536 Sep 13 23:02 ncsharp
$ _
```



Mkbundle can also cross-compile

- Many targets to choose from
- Use `--list-targets`
- Then use `--cross`

```
$ mkbundle --list-targets
Available targets locally:
    default - Current System Mono
    DEMO
Targets available for download with --fetch-target:
[...]
mono-5.14.0-debian-8-arm
mono-5.14.0-debian-8-arm64
mono-5.14.0-debian-8-armel
mono-5.14.0-debian-8-ppc64el
mono-5.14.0-debian-8-x64
mono-5.14.0-debian-8-x86
mono-5.14.0-debian-9-arm
mono-5.14.0-debian-9-arm64
mono-5.14.0-debian-9-armel
mono-5.14.0-debian-9-ppc64el
mono-5.14.0-debian-9-x64
mono-5.14.0-debian-9-x86
mono-5.14.0-osx-10.7-i386
mono-5.14.0-osx-10.7-x64
mono-5.14.0-raspbian-8-arm
mono-5.14.0-raspbian-9-arm
mono-5.14.0-ubuntu-14.04-arm
mono-5.14.0-ubuntu-14.04-arm64
mono-5.14.0-ubuntu-14.04-ppc64el
mono-5.14.0-ubuntu-14.04-x64
mono-5.14.0-ubuntu-14.04-x86
mono-5.14.0-ubuntu-16.04-arm
mono-5.14.0-ubuntu-16.04-arm64
mono-5.14.0-ubuntu-16.04-ppc64el
mono-5.14.0-ubuntu-16.04-x64
mono-5.14.0-ubuntu-16.04-x86
mono-5.14.0-ubuntu-18.04-arm
mono-5.14.0-ubuntu-18.04-arm64
mono-5.14.0-ubuntu-18.04-ppc64el
mono-5.14.0-ubuntu-18.04-x64
mono-5.14.0-ubuntu-18.04-x86

$
```

Cross compiling, and single binary ge

```
$  uname -a
Darwin iMac.local 17.7.0 Darwin Kernel Version 17.7.0: Thu Jun 21 22:53:14 PDT 2018; root:xnu-
4570.71.2~1/RELEASE_X86_64 x86_64
$ mkbundle --fetch-target mono-5.14.0-ubuntu-18.04-arm64
$ mkbundle --list-targets
Available targets locally:
Available targets locally:
        default - Current System Mono
        DEMO
        mono-5.14.0-ubuntu-18.04-arm64
$ mkbundle --cross mono-5.14.0-ubuntu-18.04-arm64 command.exe
From: /Users/miguel/.mono/targets/mono-5.14.0-ubuntu-18.04-arm64
Using runtime: /Users/miguel/.mono/targets/mono-5.14.0-ubuntu-18.04-arm64/bin/mono
     Assembly: /cvs/NetCatNetwork/command.exe
     Assembly: /Users/miguel/.mono/targets/mono-5.14.0-ubuntu-18.04-
arm64/lib/mono/4.5/mscorlib.dll
     Assembly: /Users/miguel/.mono/targets/mono-5.14.0-ubuntu-18.04-
arm64/lib/mono/4.5/I18N.West.dll
     Assembly: /Users/miguel/.mono/targets/mono-5.14.0-ubuntu-18.04-arm64/lib/mono/4.5/I18N.dll
Generated a.out
$ file a.out
a.out: ELF 64-bit LSB shared object, ARM aarch64, version 1 (SYSV), dynamically linked,
interpreter /lib/ld-linux-aarch64.so.1, for GNU/Linux 3.7.0,
BuildID[sha1]=18b7149b2df3a702db1ffb8cbe0e0508c6f5c2b5, stripped
$ _
```