Asserts, Errors, and Exceptions

Deborah Kurata
http://msmvps.com/blogs/deborahk/
@DeborahKurata
deborahk@insteptech.com

pluralsight
hardcore dev and IT training

Defensive Coding

Clean Code

- Improves Comprehension
- Simplifies Maintenance
- Reduces Bugs

Testable Code + Unit Tests

- Improves Quality
- Confirms Maintenance
- Reduces Bugs

Validation + Exception Handling

- Improves Predictability
- More Consistent
- Reduces Bugs

Topics

Asserts

Errors

Exceptions

Anticipated Issues and Exceptions

Invalid User Entry

Invalid or Missing Data

Code Construct Issues

System Issues

User Entry

Invalid User Entry

Invalid or Missing Data

Code Construct Issues

System Issues

- **Use an appropriate control**
- **Use built in data validation**
- **Write a validation method**
  - Display message to the user
- **Validate with guard clauses**
  - Display message to the user
- **Proceed with a good default value**

Invalid or Missing Data

- Invalid User Entry
- Invalid or Missing Data
- Code Construct Issues
- System Issues

- Validate in coming data
- Proceed without the value
- Proceed with a good default value
- Display a message to the user

Code Construct Issues

Invalid User Entry

Invalid or Missing Data

Code Construct Issues

System Issues

- **Proceed with a default operation**
- **Ignore the issue**
- **Log it and display a message to the user**

System Issues

Invalid User Entry

Invalid or Missing Data

Code Construct Issues

System Issues

- Try again
- Proceed with an alternate operation
- Ignore the issue
- Log it and display a message to the user

Anticipated Issues and Exceptions

Invalid User Entry

Invalid or Missing Data

Code Construct Issues

System Issues

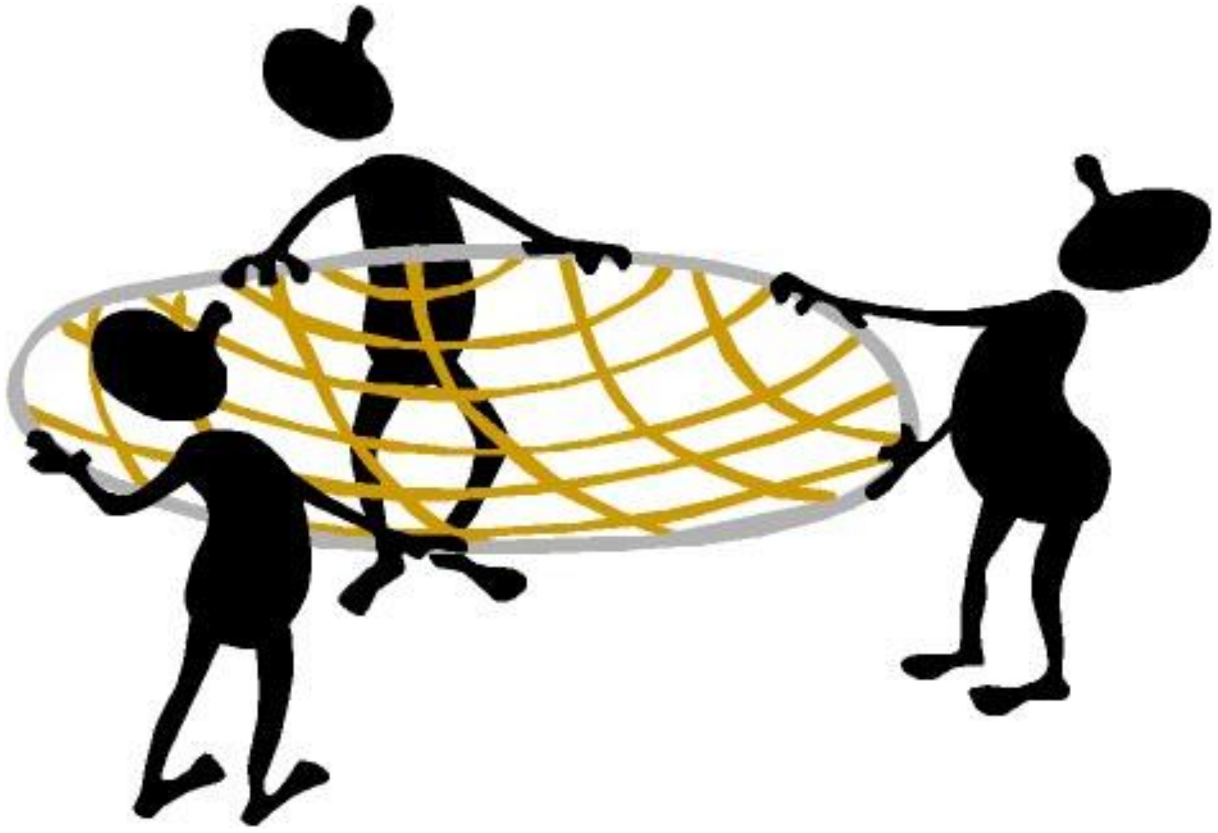
- Use restrictive controls and binding
- Use validation methods
- Use good defaults
- Return Operation Result
- Throw exceptions
- Notify the user only when necessary

Unexpected Exceptions

To expect the unexpected shows a thoroughly modern intellect

Oscar Wilde

Global Exception Handler



Summary

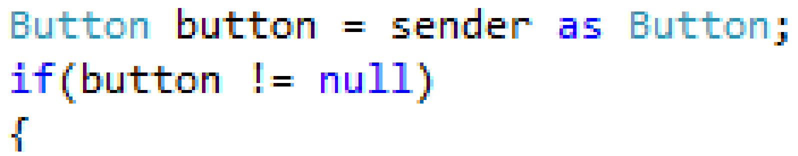
Asserts

Errors

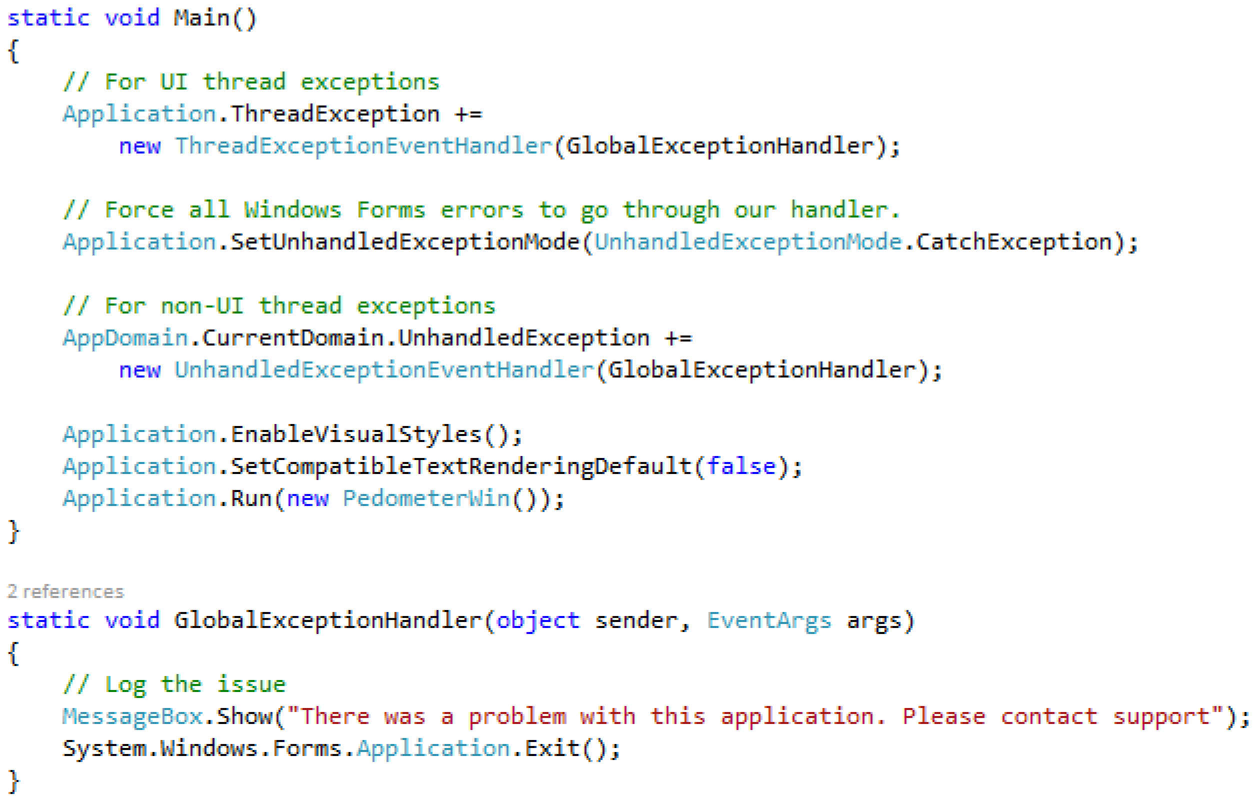
Exceptions

Preventing and Handling Errors and Exceptions

- Use controls
- Use code
- Use Debug.Assert to check Invariants
- Catch exceptions from guard clauses
- Catch exceptions from code constructs
- Catch exceptions from system issues
- Define a global exception handler



```
Button button = sender as Button;
if(button != null)
{
```



```
static void Main()
{
    // For UI thread exceptions
    Application.ThreadException +=
        new ThreadExceptionEventHandler(GlobalExceptionHandler);

    // Force all Windows Forms errors to go through our handler.
    Application.SetUnhandledExceptionMode(UnhandledExceptionMode.CatchException);

    // For non-UI thread exceptions
    AppDomain.CurrentDomain.UnhandledException +=
        new UnhandledExceptionEventHandler(GlobalExceptionHandler);

    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);
    Application.Run(new PedometerWin());
}

2 references
static void GlobalExceptionHandler(object sender, EventArgs args)
{
    // Log the issue
    MessageBox.Show("There was a problem with this application. Please contact support");
    System.Windows.Forms.Application.Exit();
}
```

Handling Anticipated Exceptions

Additional Topics

- Multiple catch blocks
- Finally clause
- Custom Exceptions

Summary