Defensive Coding in C#

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

pluralsight
hardcore dev and IT training

What is Defensive Coding?

… an approach to improve software and source code, in terms of:

- General quality - Reducing the number of software bugs and problems.
- Making the source code comprehensible - the source code should be readable and understandable so it is approved in a code audit.
- Making the software behave in a predictable manner despite unexpected inputs or user actions.

- Wikipedia as of 4/14/14

What is Defensive Coding?

Defensive programming is an approach to improve software and source code, in terms of:

- General **quality** - Reducing the number of software bugs and problems.
- Making the source code comprehensible - the source code should be readable and understandable so it is approved in a code audit.
- Making the software behave in a predictable manner despite unexpected inputs or user actions.

- Wikipedia as of 4/14/14

Automated Code Testing

What is Defensive Coding?

Defensive programming is an approach to improve software and source code, in terms of:

- General quality - Reducing the number of software bugs and problems.
- Making the source code **comprehensible** - the source code should be readable and understandable so it is approved in a code audit.
- Making the software behave in a predictable manner despite unexpected inputs or user actions.

- Wikipedia as of 4/14/14

Clean Code

What is Defensive Coding?

Defensive programming is an approach to improve software and source code, in terms of:

- General quality - Reducing the number of software bugs and problems.
- Making the source code comprehensible - the source code should be readable and understandable so it is approved in a code audit.
- Making the software behave in a **predictable** manner despite unexpected inputs or user actions.

- Wikipedia as of 4/14/14

Validation + Exception Handling

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

Clean Coding

Good code vs Bad code

Write Good code following clean coding techniques

Transform Bad code to Good code through refactoring

Clean Code

- Easy to read
- Clear intent
- Simple
- Minimal
- Thoughtful

The Code

Refactoring

Refactoring

Refactoring

Refactoring

Defensive Coding

Clean Code

- Improves Comprehension
- Simplifies Maintenance
- Reduces Bugs

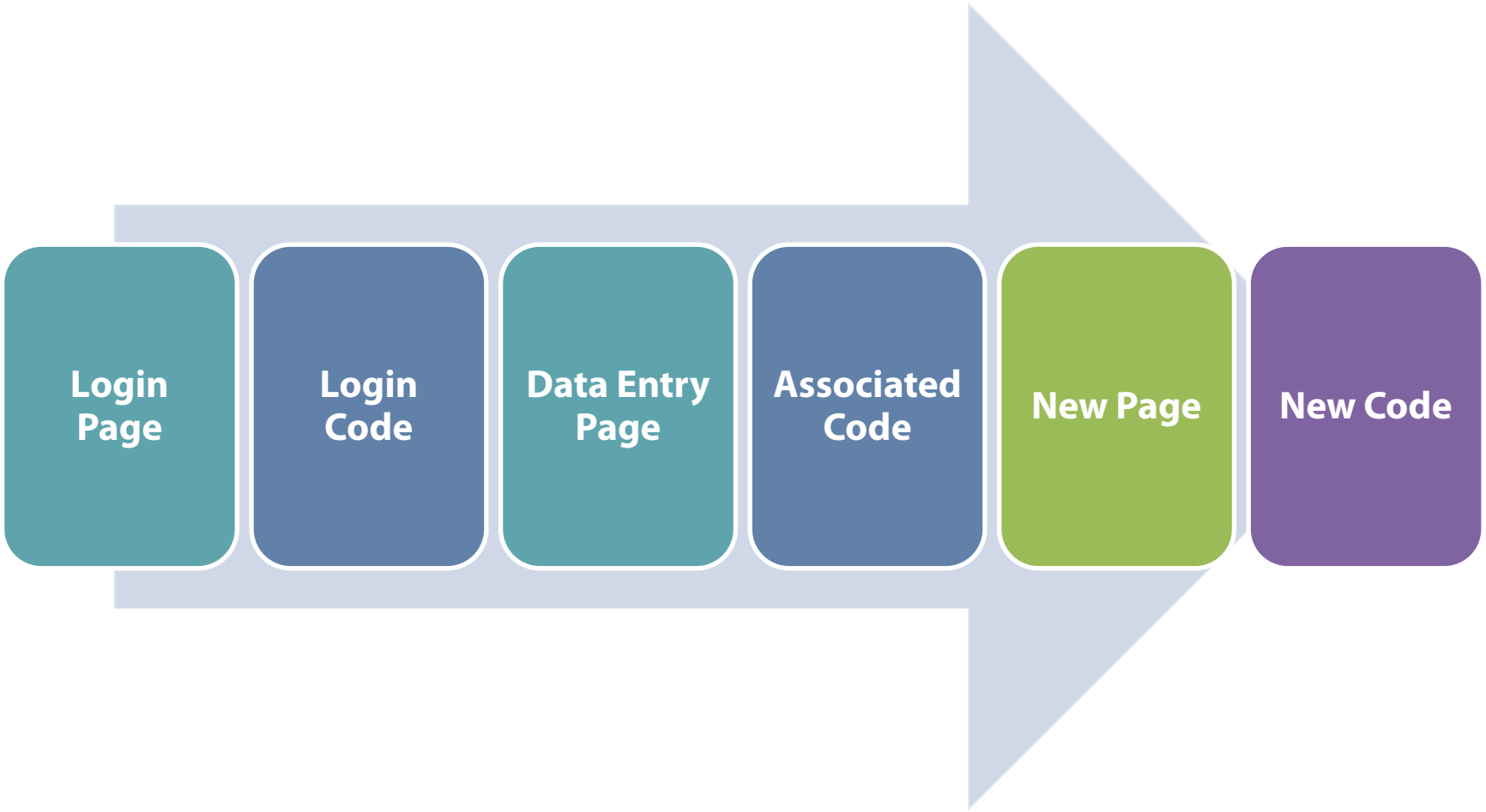
Testable Code
+
Unit Tests

- Improves Quality
- Confirms Maintenance
- Reduces Bugs

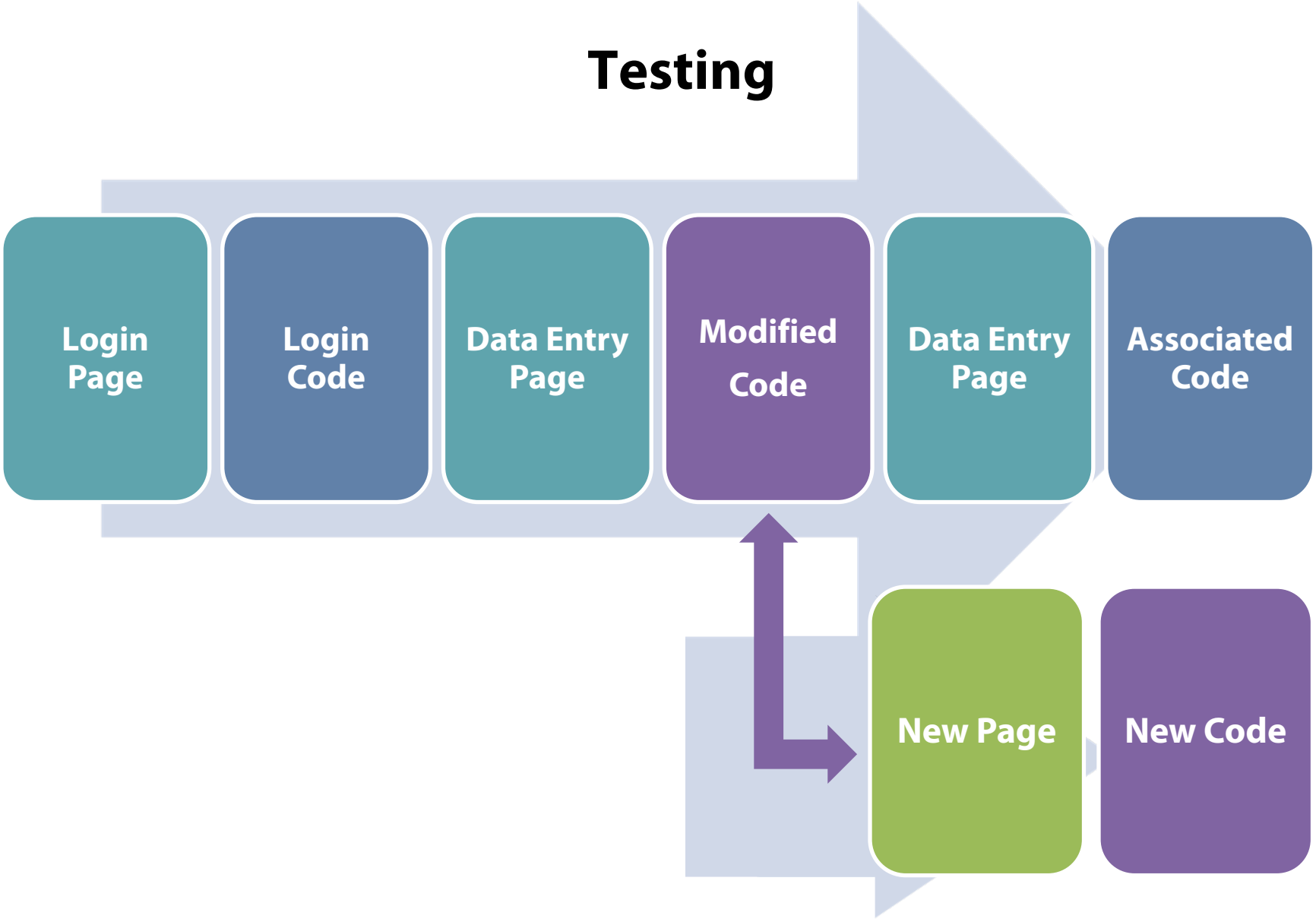
Validation
+
Exception Handling

- Improves Predictability
- More Consistent
- Reduces Bugs

Testing


Login Page
Login Code
Data Entry Page
Associated Code
New Page
New Code

Testing


Login Page
Login Code
Data Entry Page
Modified Code
Data Entry Page
Associated Code
New Page
New Code

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

Trust but Verify

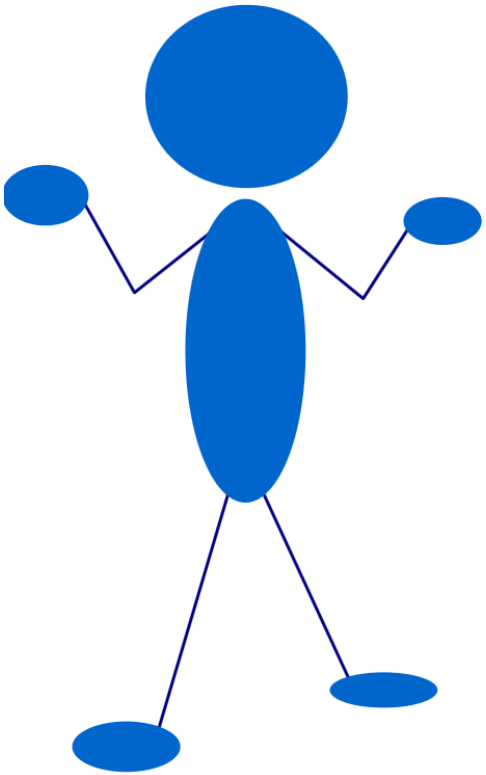
Contract

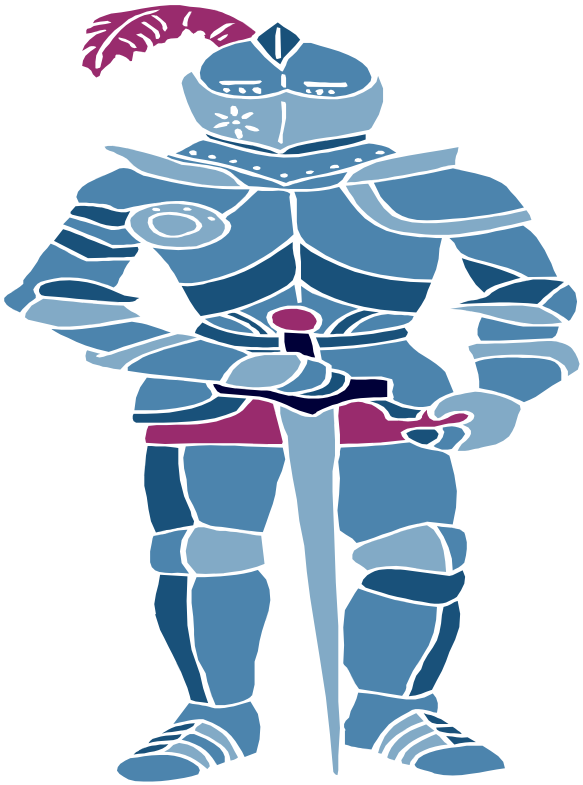
- Parameters
- Return Type
- Exceptions

Verify

- Parameters
- Data
- Return Type
- Exceptions

Clean yet Protected





Topics

- Defending Your Methods - Part 1
- Defending Your Methods Part 2: Validating Method Parameters
- Automated Code Testing
- Defending Your Methods Part 3: Returning Predictable Results
- Defending Your Code Constructs
- Asserts, Errors and Exceptions
- Final Words

Clean Code

Testable Code
+
Unit Tests

Validation
+
Exception Handling