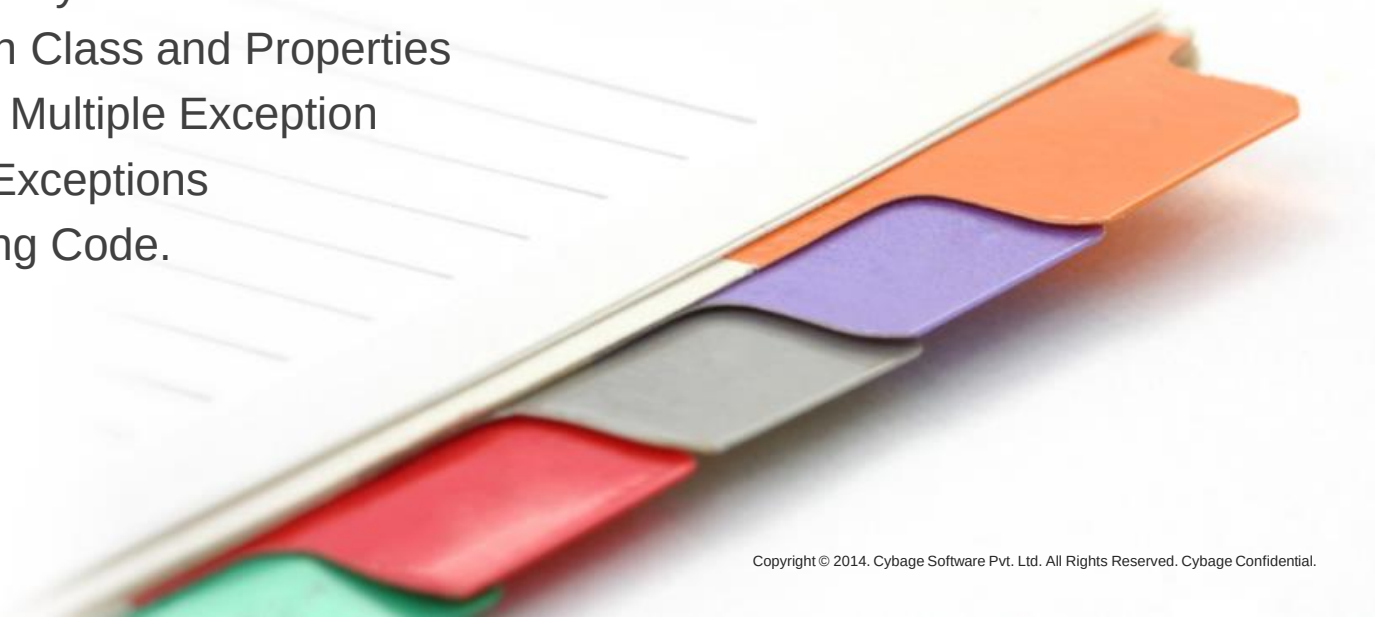
Exception Handling using C#

Sushant Banerjee | sushantba@cybage.com | 7221

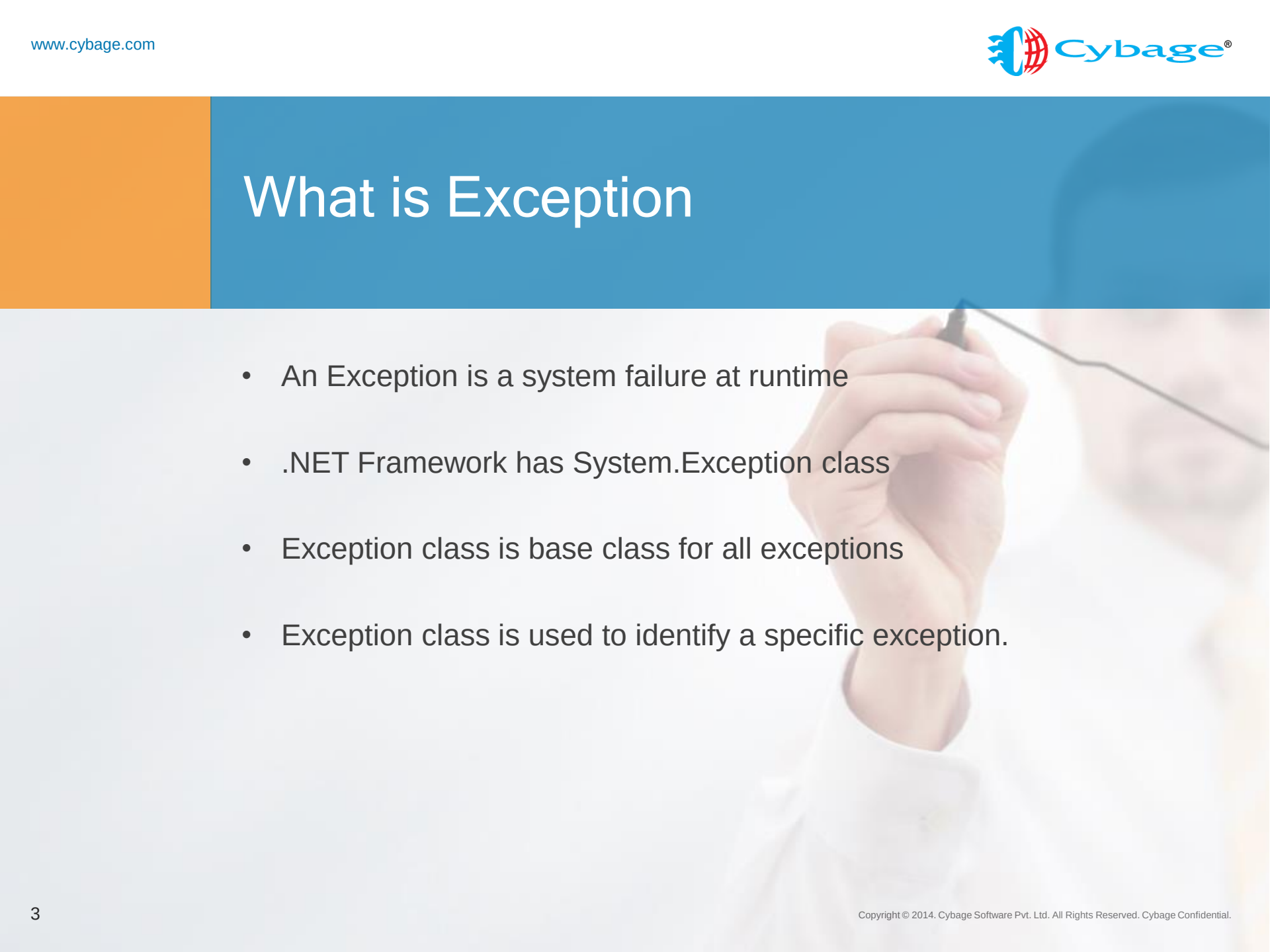
Agenda

- Introduction to Exceptions
- Handling Exceptions
- Using Try..Catch Block
- Exception Bubbling
- Using Finally Block
- Exception Class and Properties
- Handling Multiple Exception
- Custom Exceptions
- Debugging Code.



What is Exception

- An Exception is a system failure at runtime
- .NET Framework has System.Exception class
- Exception class is base class for all exceptions
- Exception class is used to identify a specific exception.



What is Exception Handler

- When an exception is thrown
  - Known as unhandled exceptions
  - Program terminates
- Write code to catch the exception thrown
  - Known as Exception Handler
  - Appropriate action is taken.

Try…Catch Block

- Role of try block
  - Contains code that may throw exceptions
  - When an exception is thrown runtime looks for a catch block
- Role of catch block
  - Contains code that handles exceptions
  - Catch block decides what to do next.

Exception Bubbling

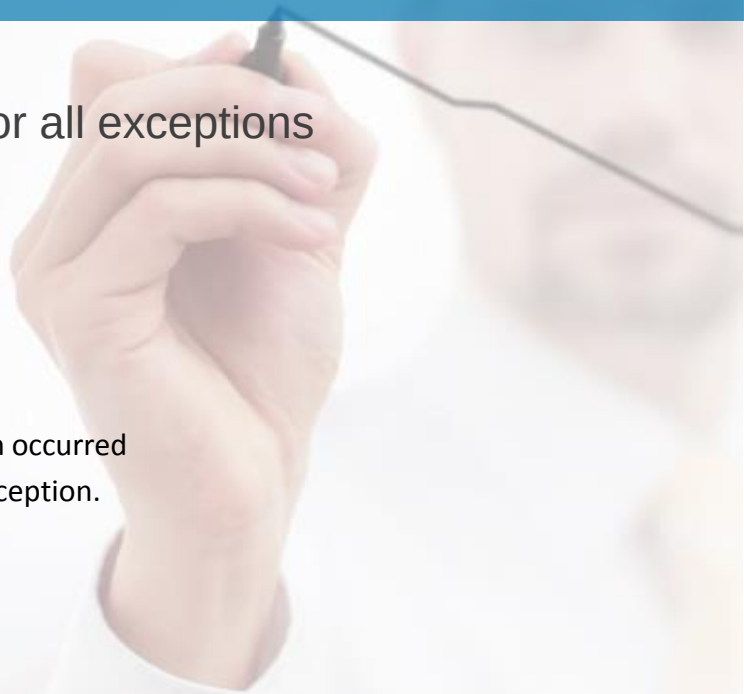
- Nested method calls
  - Method1 calls Method2 inside try block
  - Method2 calls Method3 without try block
  - Method3 throws an exception
- What runtime does
  - Runtime jumps back on the call stack
  - Since there is no handler in Method2
  - Exception is handled in Method1
  - Known as Exception Bubbling.

Finally Block

- Some code never execute in try block
  - Code written after the line which throws exception
  - That code may be releasing resources and cleaning memory
  - So, it is important to execute that code but how?
- Finally block
  - Executes whether an exception is thrown or not
  - Write code here to cleanup memory.

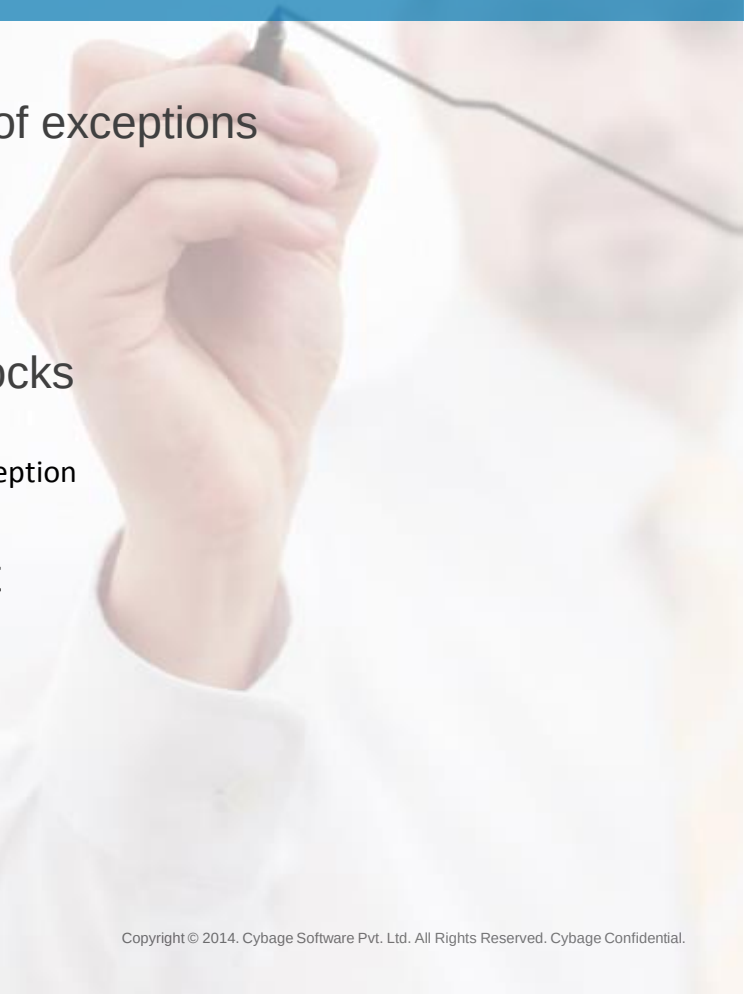
Exception Class

- System.Exception is the base class for all exceptions
  - Such as DivideByZeroException
- Properties of exception object
  - Message – describe the current exception
  - Source – name of the application caused exception
  - StackTrace – shows method call stack where exception occurred
  - TargetSite – shows method that throws the current exception.



Handling Multiple Exceptions

- A try block may throw different types of exceptions
  - DivideByZeroException
  - FileNotFountException
  - IndexOutOfRangeException
- A try block may use multiple catch blocks
  - Runtime searches for a specific catch
  - A catch with matching exception type handles the exception
- Order of the catch blocks is important
  - A more specific catch should come first
  - A generic catch should come last.



Throwing Custom Exceptions

- Custom exceptions
  - When you detect a problem
  - You may consider throwing exception

```
class CustomException : ApplicationException
{
    public CustomException(string message) : base(message)
    {
    }
}
```

```
if(salary == 0)
{
    throw new CustomException("Salary cannot be zero");
}
```

Demo

Debugging

- Setting break point
- Using step by step execution
- Looking at values inside variables
- Skipping method execution

Demo

Bibliography, Important Links

http://msdn.microsoft.com/en-us/library/ms173160(v=vs.110).aspx

http://www.codeproject.com/Articles/125470/Exception-Handling-for-C-Beginners

Any Questions?

Thank you!