Code Quality

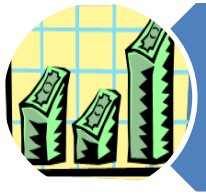
Steigerung der Codequalität mit
Visual Studio & TFS

Tobias Richling

- 30 Jahre
- Wohnhaft im Münsterland
- Softwareentwickler
  seit Amiga Basic
- „Microsoftie“
- Logisitksoftware
  - Silverlight
  - TFS
- Trainer, Speaker, Autor

Agenda

Clean Code



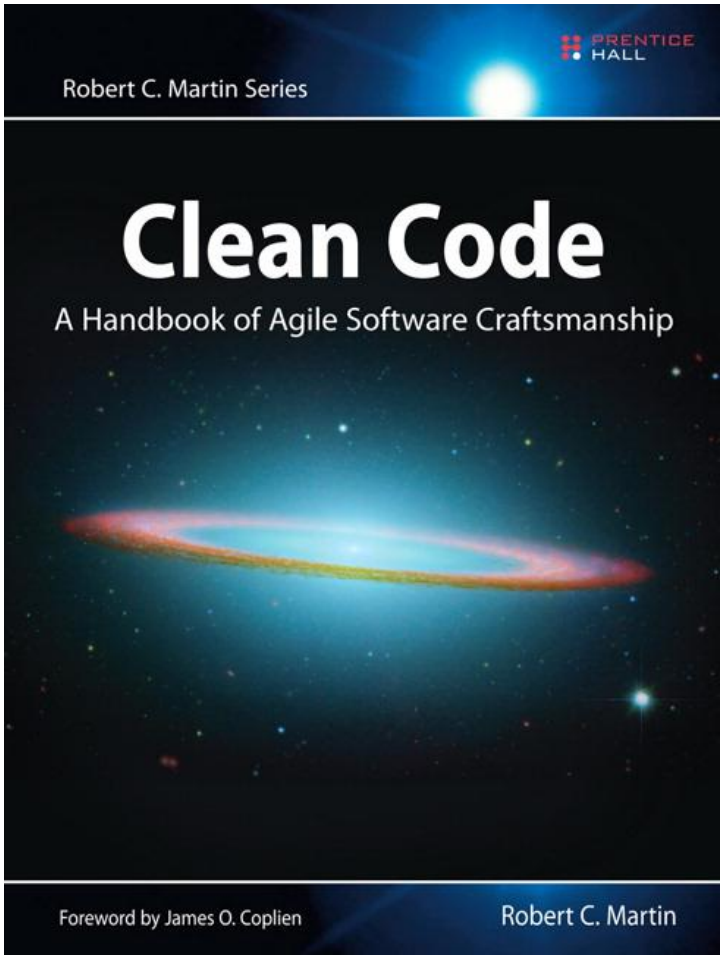
Code Analysis

Checkin Policies

Ausblick: Roslyn

CLEAN CODE

Wer kennt dieses Buch?



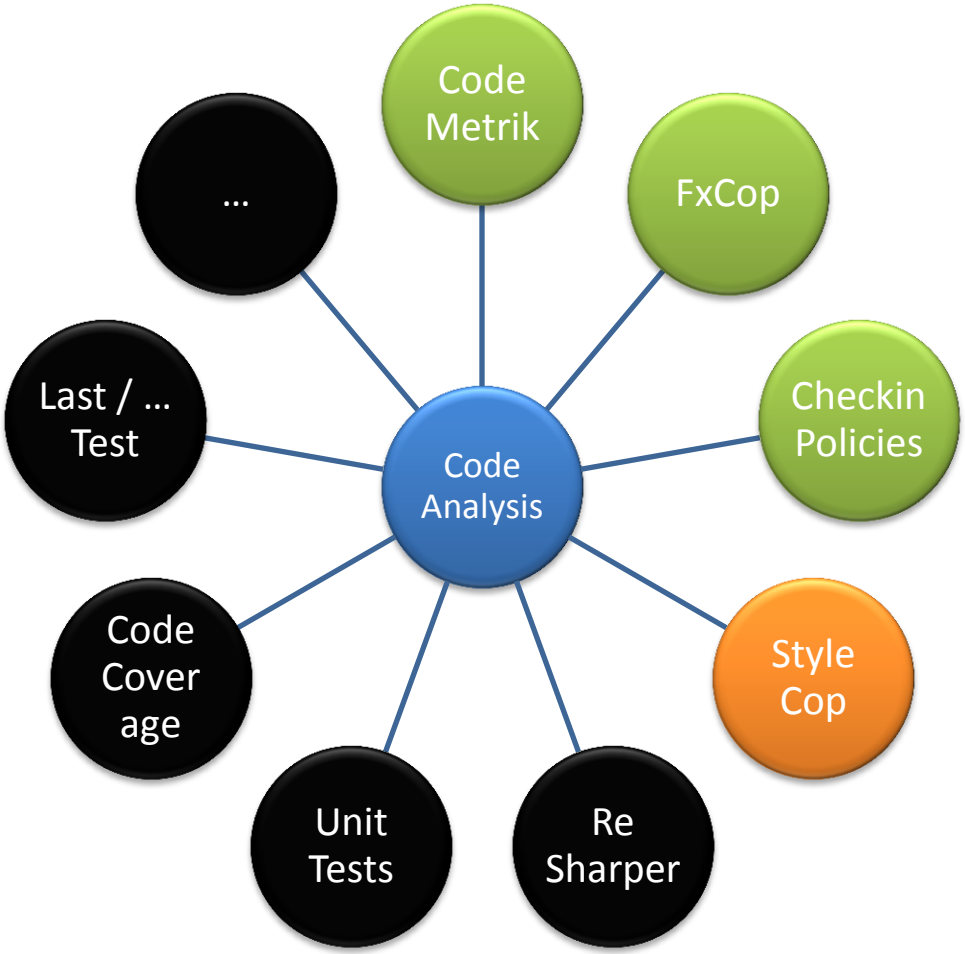
„Uncle Bob“

Einige Beispiele für Clean Code

- Funktionen
  - Einheitliches Abstraktionsniveau
  - KURZ!
- Law of Demeter
  - Keine Referenzketten
  - Schüchterner Code
- Tell dont ask
  - Öffentliche Properties vermeiden

CODE ANALYSIS

Code Analysis ist ein weites Feld



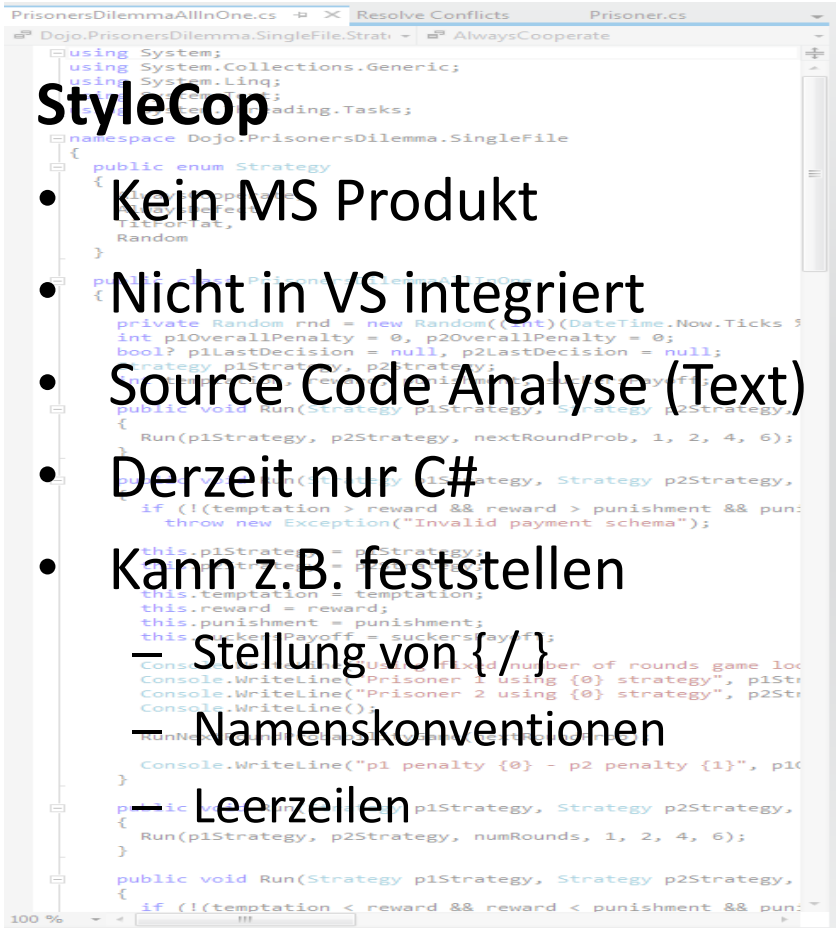
Einige Spielarten der Code Analyse

- Code Metriken
  - Berechnung von Kennzahlen über den Code
- Visual Studio Code Analysis
  - Prüfung von Architektur- / Sicherheits-Konventionen
  - basiert auf FxCop
- Checkin Policies
  - Prüft Dateien / Formalien beim Checkin
- StyleCop
  - Prüfung von Code-Konventionen

Code Metriken

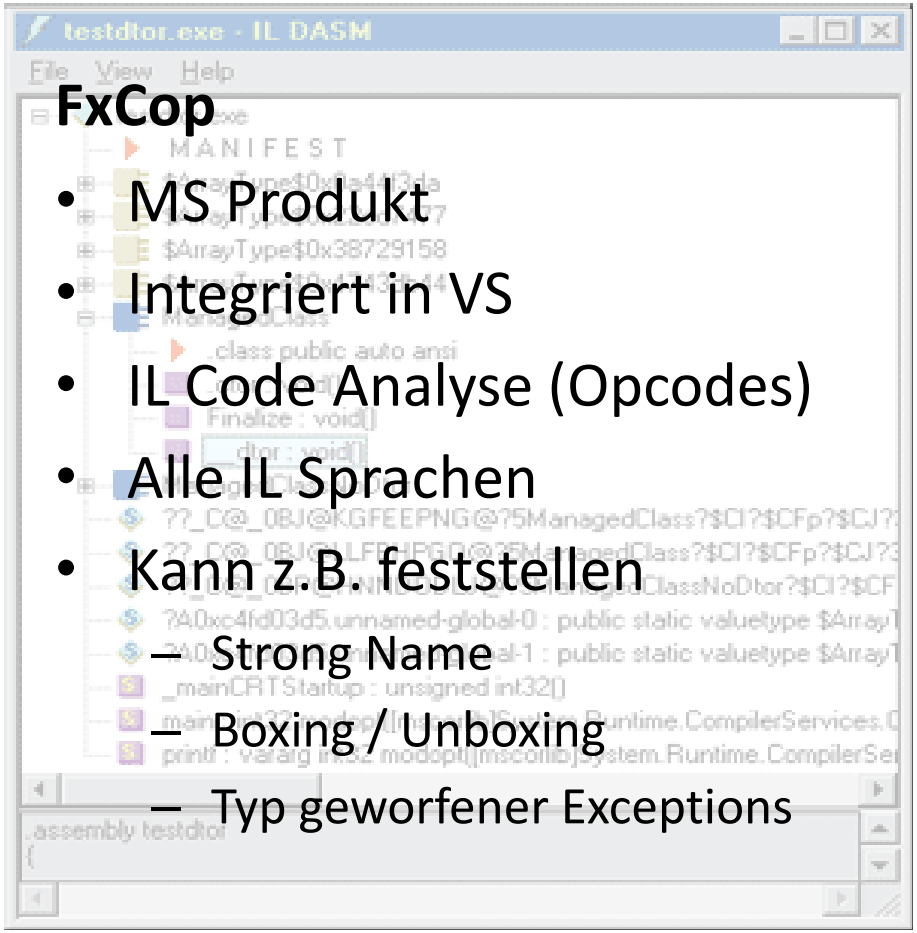
- Maintainability Index
  - Zusammengesetzter Wert
- Cyclomatic Complexity
  - Wie viele Pfade gibt es durch eine Methode
- Depth of inheritance
  - Vererbungstiefe
- Class coupling
  - Wie viele Klassen werden verwendet
- Lines of code
  - Codezeilen ohne Kommentare

StyleCop vs. FxCop



StyleCop

- Kein MS Produkt
- Nicht in VS integriert
- Source Code Analyse (Text)
- Derzeit nur C#
- Kann z.B. feststellen
  - Stellung von { / }
  - Namenskonventionen
  - Leerzeilen

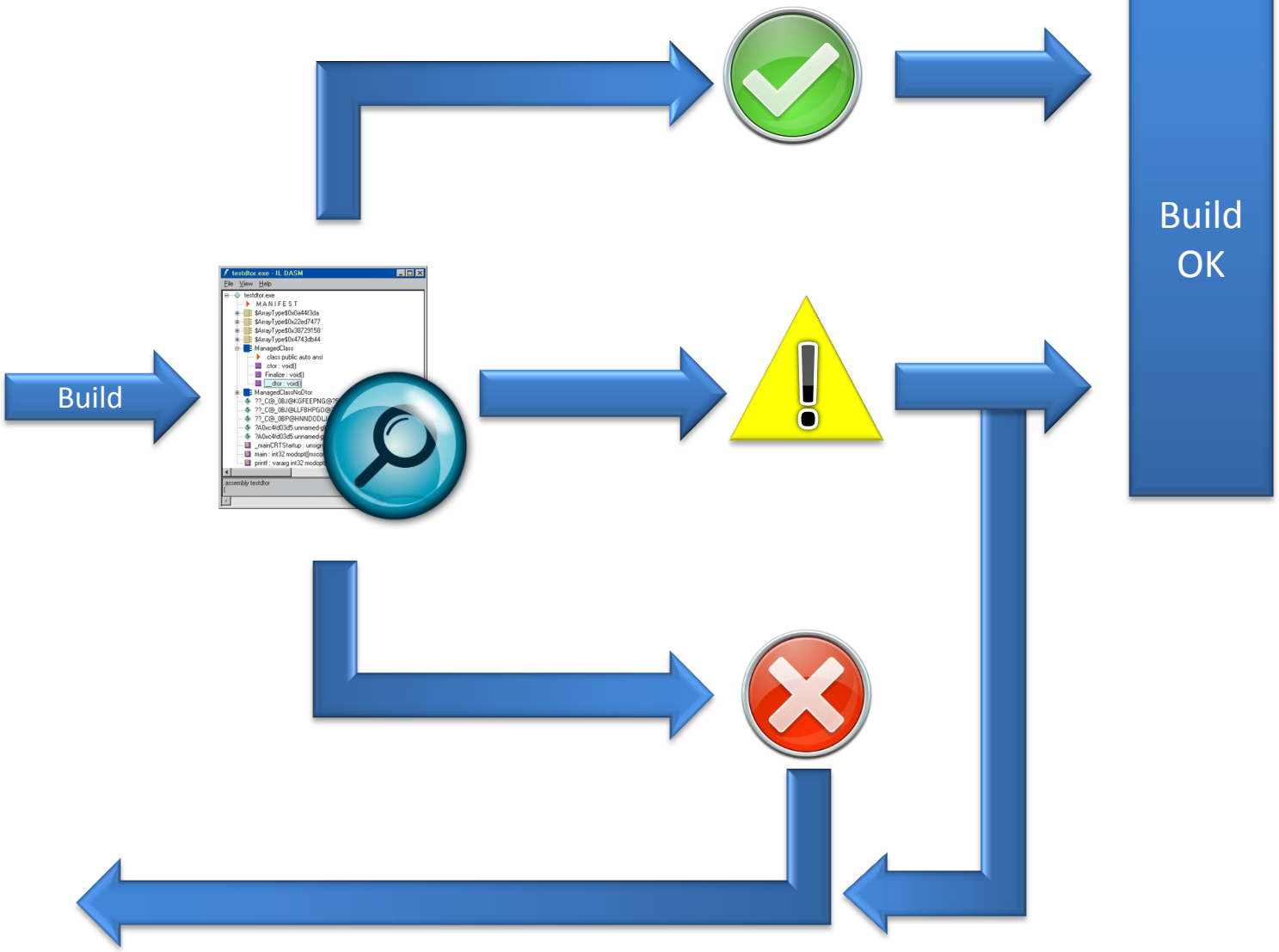


FxCop

- MS Produkt
- Integriert in VS
- IL Code Analyse (Opcodes)
- Alle IL Sprachen
- Kann z.B. feststellen
  - Strong Name
  - Boxing / Unboxing
  - Typ geworfener Exceptions

Es gibt Überschneidungen!

FxCop - Ablauf



Out of the box

- FxCop seit VS2010 teil des Visual Studio
  - C:\Program Files (x86)\Microsoft Visual Studio 11.0\Team Tools\Static Analysis Tools\FxCop
- Große Anzahl an vorgefertigten Regeln
  - Thematisch gruppiert
- Einige Regeln für den Anfang auswählen
  - Sonst kann man „erschlagen“ werden

Beispielhafte Coderegeln

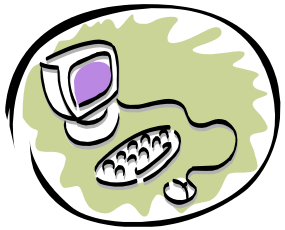
- Übermäßige Komplexität vermeiden
  - Zyklomatische Komplexität
- Ungenutzte Methoden entfernen
- Ungenutzte Felder entfernen
- Methoden statisch machen
  - Falls Methode keine Klassenmember nutzt, sollte diese statisch definiert werden

Eigene Regeln entwicklen

- Ableiten von BaseIntrospectionRule
- Überschreiben der Check-Methode
  - Untersuchung von Typen, Membern etc.
  - Reflection-Level analyse
- Überschreiben der Visit-Methode
  - Untersuchung des IL Codes
  - Introspection
  - Visitor-Pattern

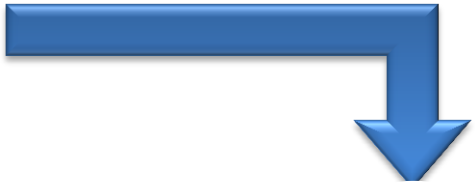
CHECKIN POLICIES

Ablauf



Client

Server



Checkin

Policy

Override

Warum?

Out of the Box

- Eine Handvoll Policies werden mit geliefert
  - Work Item
  - Code Comment
- Die TFS Power Tools bringen einige weitere
  - Work Item Query
  - Custom Path
- Diese kann man auch mit TFS 11 Beta verwenden

Eigene Checkin Policies entwicklen

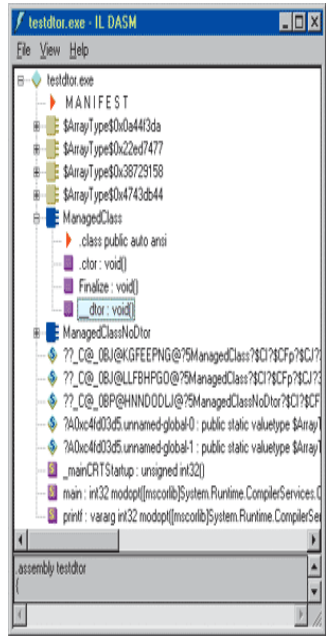
- Implementiere
  - IPolicyDescription & IPolicyExecution oder
  - Erben von PolicyBase
- Deployment auf jeden Client!
  - Z. B. einchecken im TFS
- Registrieren in der Registry

ROSLYN

Compiler Pipeline

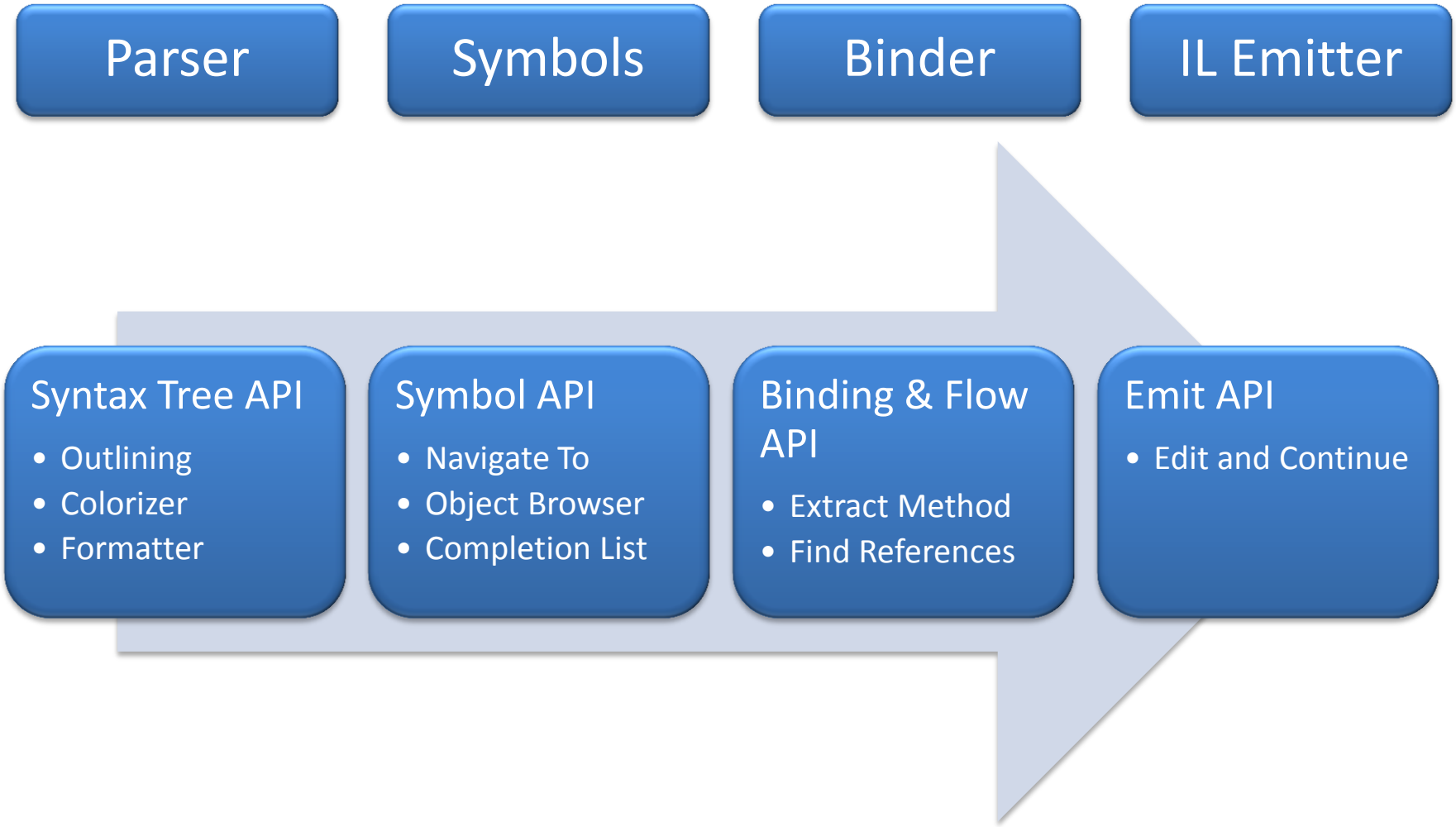
Compiler

(Black Box)



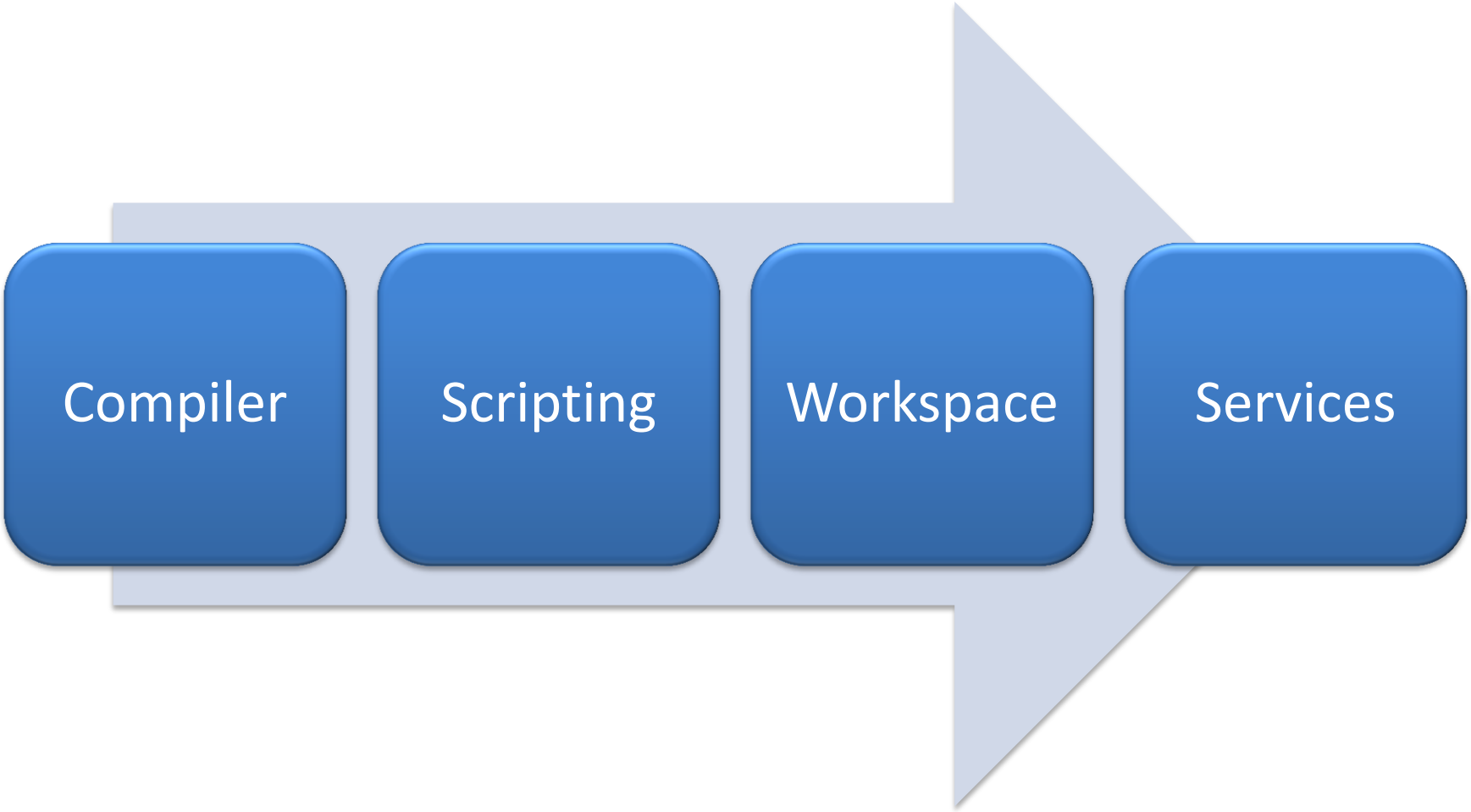
http://msdn.microsoft.com/en-us/vstudio/hh500769

Compiler API's



http://msdn.microsoft.com/en-us/vstudio/hh500769

Roslyn API Layers



Was geht?

- Code Analyse „on the fly“
  - a la Resharper
- Automatische Korrektur von Code Issues
  - a la Visual Studio Smart Tag
- Eigene Refactorings bauen

FAZIT

Fazit

- Code Analysis
  - „Code Knigge“ – unerlaubtes kritisieren
  - API schlecht dokumentiert
- Checkin Policies
  - „Türsteher“ – unerlaubtes nicht rein lassen
  - Deployment ist umständlich
- Roslyn
  - Cool! C# Interactive Window!
  - Abwarten und Tee trinken :-)

Noch Fragen

Vielen Dank für Ihre Aufmerksamkeit