Rainer Stropek | cubido

Baumschule Expression Trees in C#, CLR und DLR

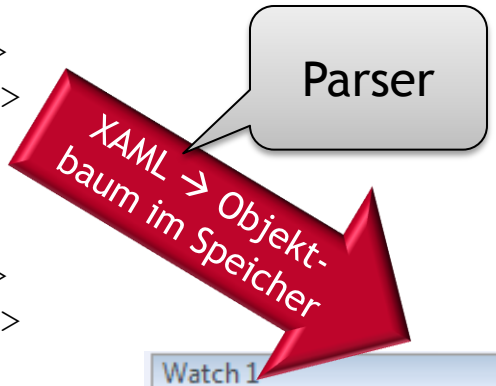
Inhalt

Als Anwendungsentwickler kommt man selten mit Expression Trees in Berührung. Sie sind ein Implementierungsdetail von LINQ. Ihre Bedeutung nimmt jedoch durch die zunehmende Beliebtheit von dynamischen Sprachen auf Basis der DLR zu. Rainer Stropek zeigt in seiner Session, was hinter Expression Trees steckt. Wie werden sie von LINQ genutzt und warum sind sie für C# 4.0, IronPython und Co. so wichtig?

SEHEN WIR DEN WALD VOR LAUTER BÄUMEN NOCH?

Von Text zum Baum

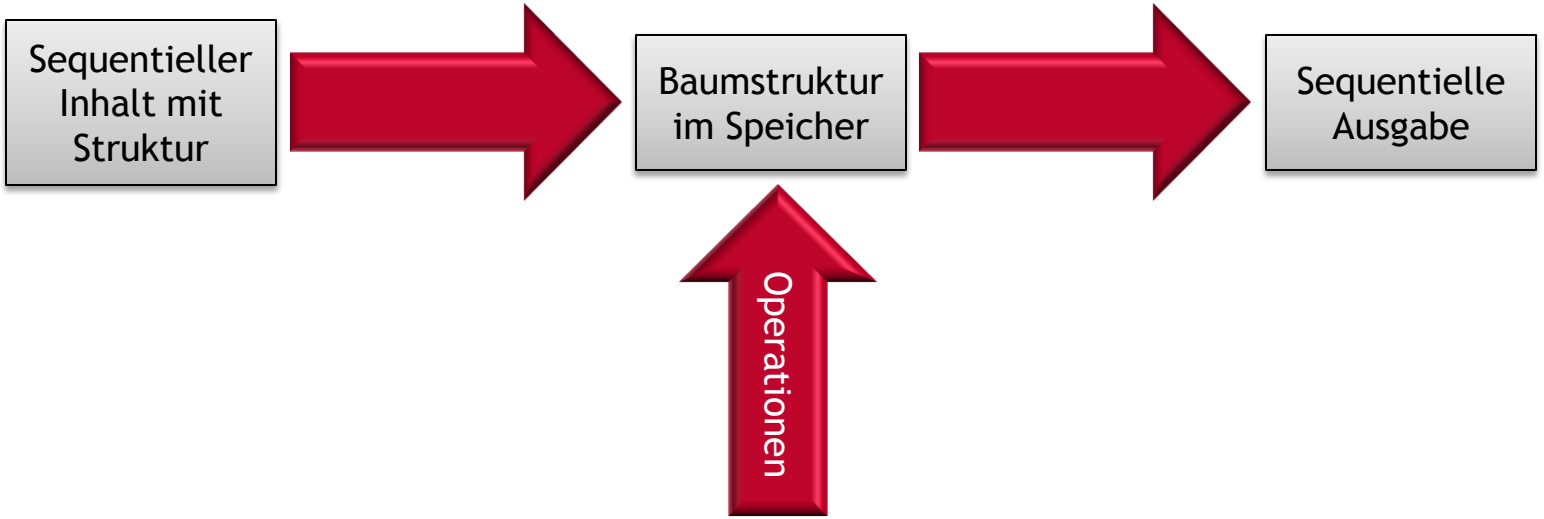
```
<Garden xmlns="clr-namespace:TreeNursery.Xaml;assembly=TreeNursery">
    <Garden.Trees>
        <Tree>
            <Tree.Fruit>
                <Apple />
            </Tree.Fruit>
        </Tree>
        <Tree>
            <Tree.Fruit>
                <Apple />
            </Tree.Fruit>
        </Tree>
        <Tree>
            <Tree.Fruit>
                <Apricot />
            </Tree.Fruit>
        </Tree>
    </Garden.Trees>
</Garden>
```



Watch 1

Name	Value
myGarden	{TreeNursery.Xaml.Garden}
Trees	Count = 3
[0]	{TreeNursery.Xaml.Tree}
Fruit	{Apple}
[TreeNursery.Xaml.Apple]	{Apple}
[1]	{TreeNursery.Xaml.Tree}
Fruit	{Apple}
[TreeNursery.Xaml.Apple]	{Apple}
[2]	{TreeNursery.Xaml.Tree}
Fruit	{Apricot}
[TreeNursery.Xaml.Apricot]	{Apricot}
Raw View	

Von Text zum Baum



Einige Beispiele

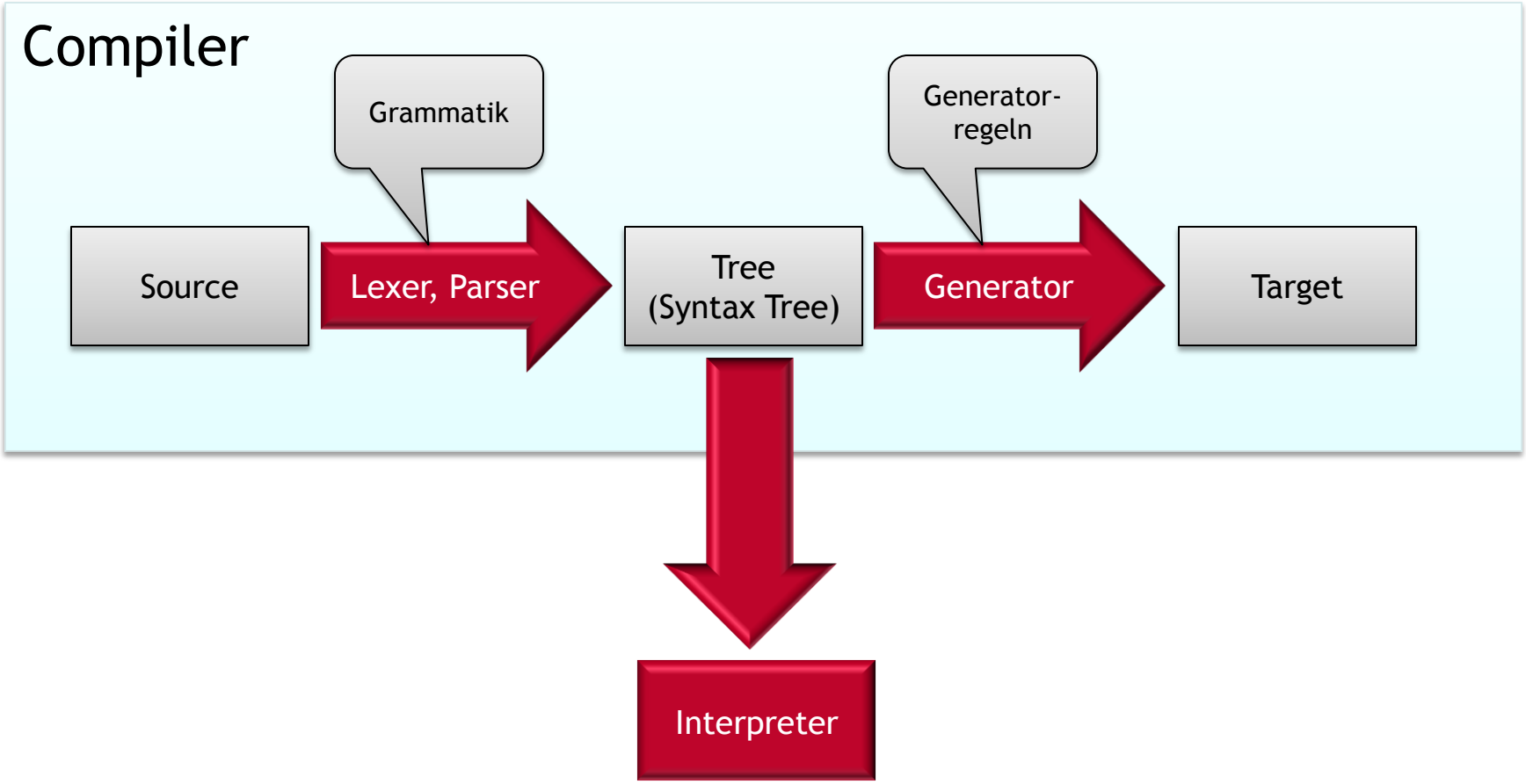
- Lexer/Parser
  - XML in DOM
  - SQL in Execution Plan
- Compiler bzw. Lexer/Parser/Generator
  - C# in IL
  - FetchXML in SQL (MS CRM)
- Interpreter
  - SQL Server Execution Plan
- Compiler-Compiler
  - ANTLR
  - Coco/R

Wichtige Begriffe

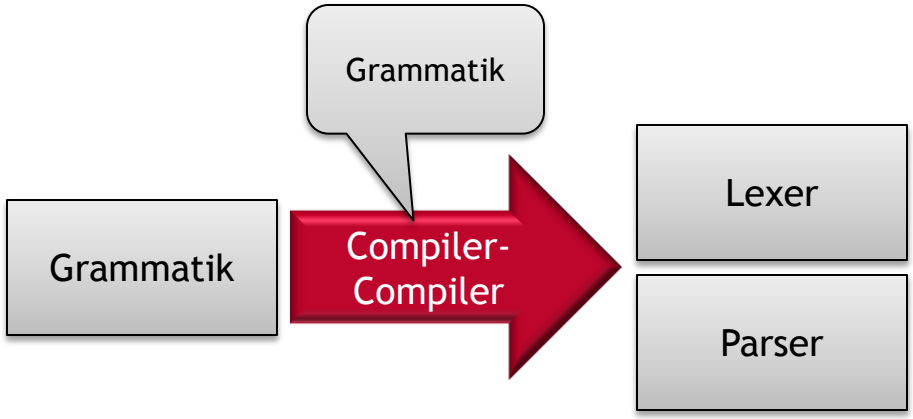
Praktisches Anwendungsbeispiel ANTLR

EIN WENIG THEORIE…

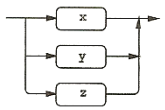
Wichtige Begriffe



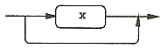
Wichtige Begriffe



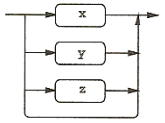
EBNF



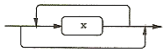
(«X»|«Y»|«Z»)
Match any alternative within the subrule exactly once.



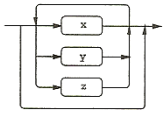
x?
Element *x* is optional.



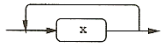
(«X»|«Y»|«Z»)?
Match nothing or any alternative within subrule.



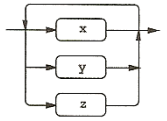
*x**
Match element *x* zero or more times.



(«X»|«Y»|«Z»)*
Match an alternative within subrule zero or more times.



x+
Match element *x* one or more times.



(«X»|«Y»|«Z»)+
Match an alternative within subrule one or more times.

Figure 4.3: EBNF GRAMMAR SUBRULES WHERE «...» REPRESENTS A GRAMMAR FRAGMENT

Quelle: The Definitive ANTLR Reference, Terence Parr

Praktisches Beispiel

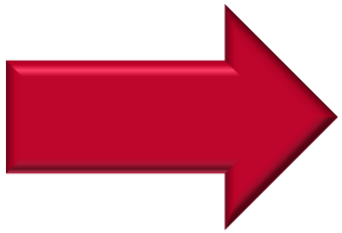
Einfache ANTLR Grammatik zum Parsen der XAML Datei

```
grammar XmlLanguage2;
options { output = AST; }

// PARSER ---------------------------------
xmlDocument : node;
node
    : '<'! ELEMENTNAME attributeList '>'!
        ( node )*
      '</'! ELEMENTNAME '>'!
    | '<'! ELEMENTNAME '/>'!;

attributeList : attribute*;
attribute : ELEMENTNAME '='! LITERAL;

// LEXER ----------------------------------
ELEMENTNAME
    : IDENTIFIER ( '.' IDENTIFIER )?;
LITERAL
    : '\'' ( ~'\'' )* '\'';
fragment IDENTIFIER
    : ( 'a'..'z' | 'A'..'Z' | '_' ) ( 'a'..'z' | 'A'..'Z' | '0'..'9' )*;
NEWLINE
    : ('\r'? '\n')+ { $channel = HIDDEN; };
WHITESPACE
    : ( '\t' | ' ' )+ { $channel = HIDDEN; } ;
```



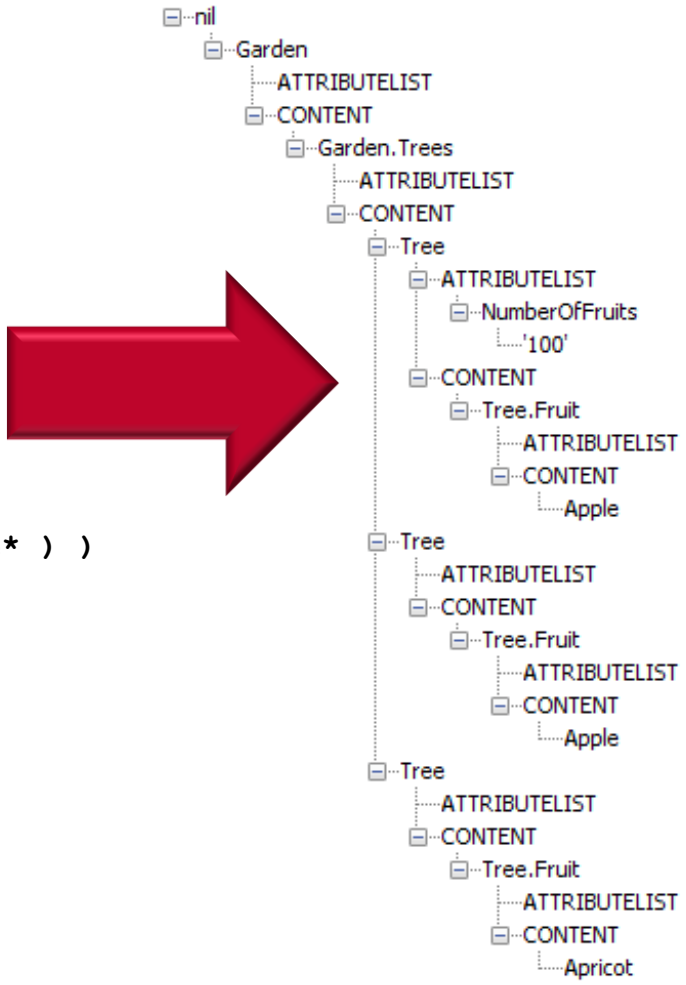
Praktisches Beispiel

Einfache ANTLR Grammatik zum Parsen der XAML Datei

```
grammar XmlLanguage;
options { output = AST; }
tokens {
    NODE = 'Node';
    ATTRIBUTELIST = 'AttributeList';
    ATTRIBUTE = 'Attribute';
    CONTENT = 'CONTENT';
}

// PARSER ---------------------------------
xmlDocument
    : node;
node
    : '<' start=ELEMENTNAME attributeList '>' ( node )*
      '</' end=ELEMENTNAME '>'
        -> ^( NODE [$start] attributeList ^( CONTENT node* ) )
    | '<' tag=ELEMENTNAME '/>'
        -> ^( NODE [$tag] );
attributeList
    : attribute*
        -> ^( ATTRIBUTELIST attribute* );
attribute
    : attribName=ELEMENTNAME '=' LITERAL
        -> ^( ATTRIBUTE [$attribName] LITERAL );

// LEXER ---------------------------------
[…]
```



Implementieren einer einfachen Formelsprache mit ANTLR

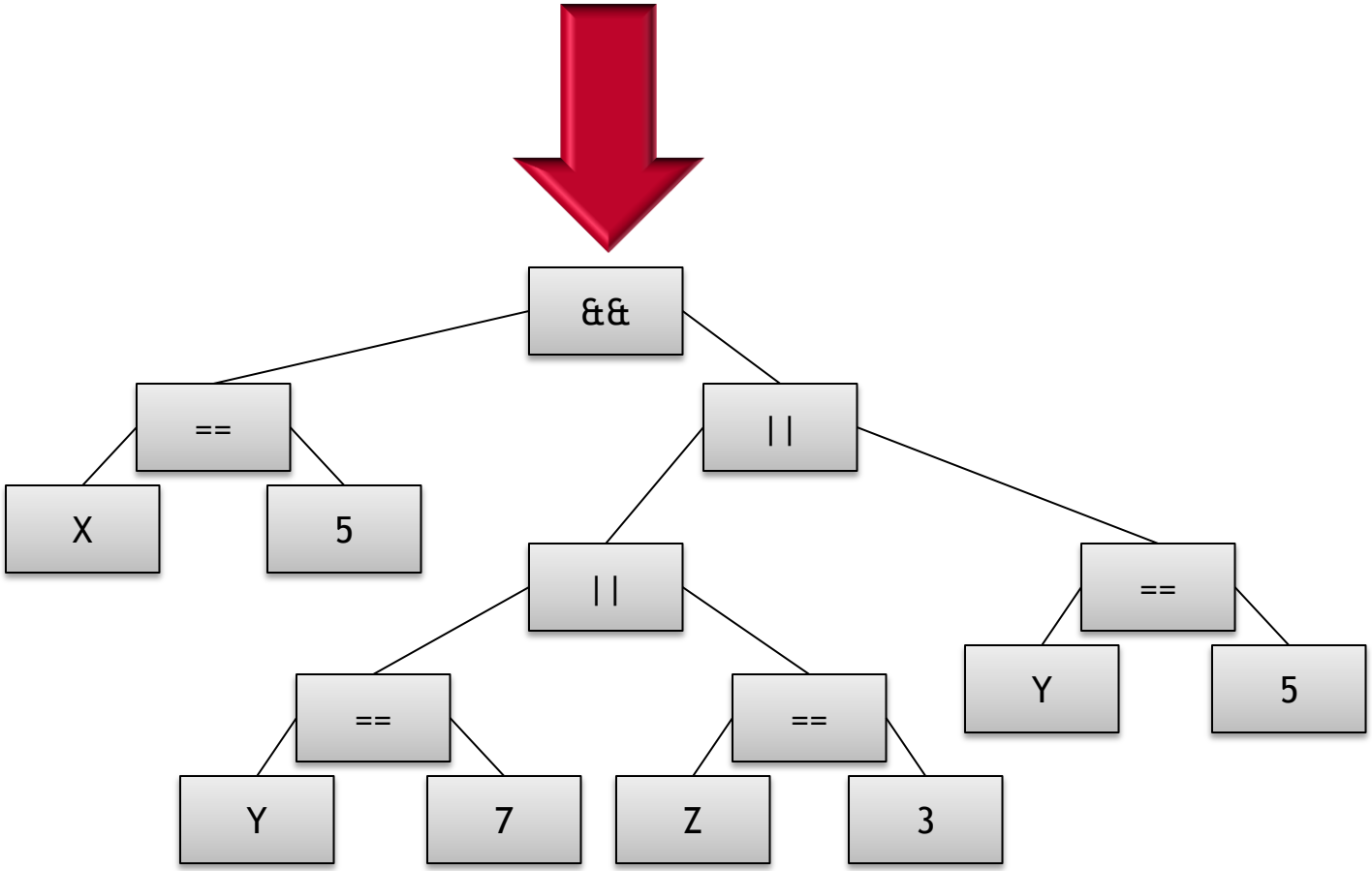
ALLES NUR THEORIE?

Praktisches Beispiel

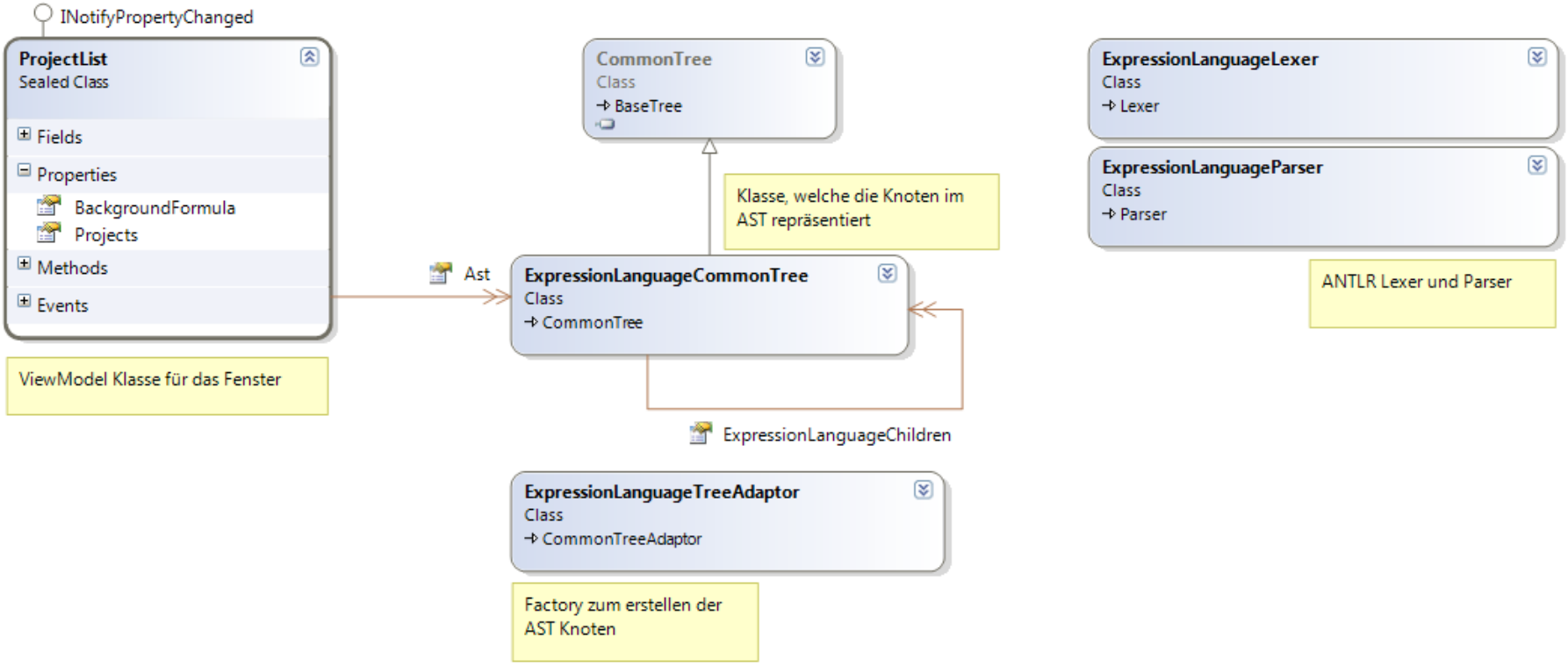
- Entwicklung einer Formelsprache zur Erweiterung einer Anwendung
- Wir unterstützen…
  - Logische Verknüpfungen
    - And, Or
  - Vergleichsoperatoren
    - =, <>, <, >, <=, >=
  - Funktionen
    - Iif
  - Zugriff auf Eigenschaften
  - Literale
    - Strings, Boolean, Numeric
- Beispiel
  - `X=5 And ( Y=7 Or Z=3 Or Y=5 )`

Wo ist der Baum?

```
X=5 And ( Y=7 Or Z=3 Or Y=5 )
```



Praktisches Beispiel



Praktisches Beispiel

Parsen der Formelsprache mit ANTLR

```
public void ParseBackgroundFormula()
{
    var stream = new ANTLRStringStream(this.BackgroundFormula);
    var lexer = new ExpressionLanguageLexer(stream);
    var tokens = new CommonTokenStream(lexer);
    var parser = new ExpressionLanguageParser(tokens);
    parser.TreeAdaptor = new ExpressionLanguageTreeAdaptor();
    this.Ast = new [] { parser.expression().Tree as ExpressionLanguageCommonTree };
}
```

Microsoft Expression Trees

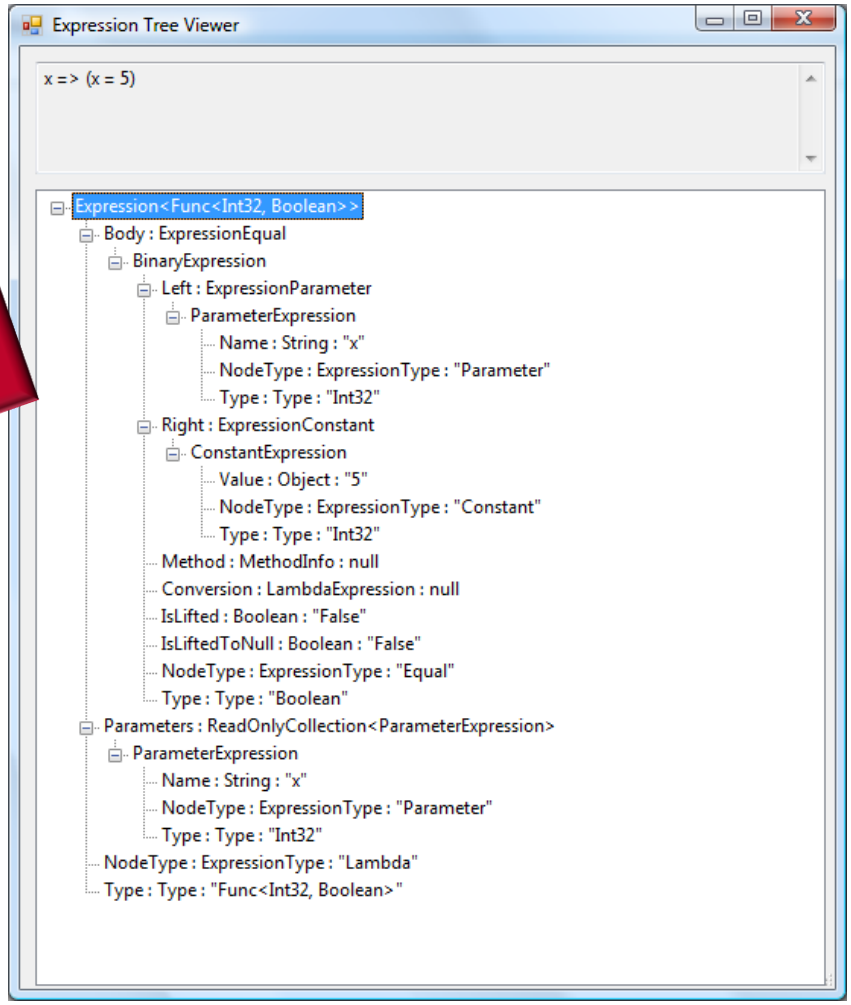
AST IN C#

ExpressionTrees in C#

Lambda Expressions vs. Expression Trees

```
Func<int, bool> f =
    (x) => x==5;

Expression<Func<int, bool>> ex =
    (x) => x == 5;
```



Expression Trees in C#

Generierter Code

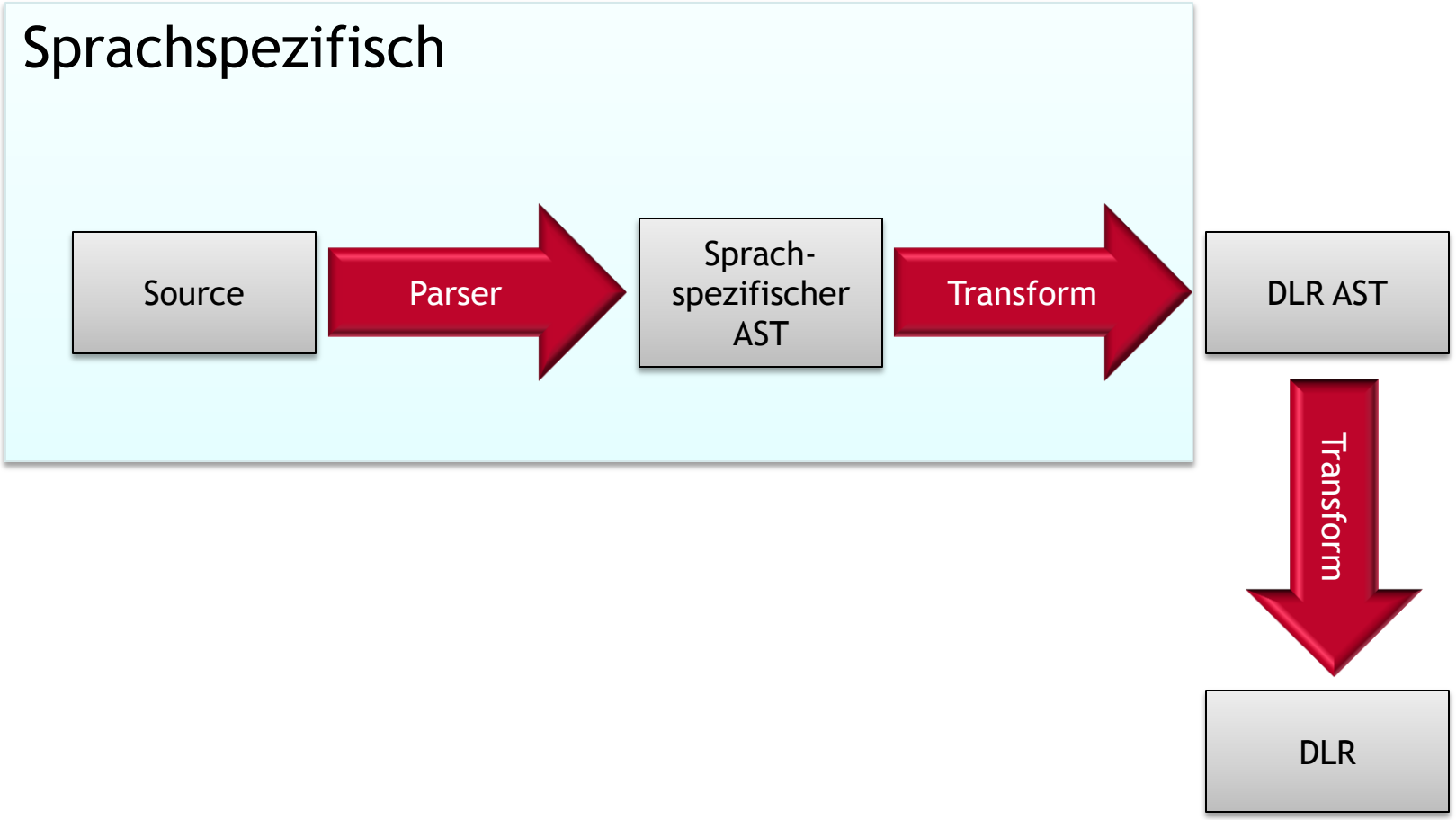
```
private static void Main(string[] args)
{
    Func<int, bool> f;
    Expression<Func<int, bool>> ex;
    […]

    ex = Expression.Lambda<Func<int, bool>>(
        Expression.Equal(
            CS$0$0000 = Expression.Parameter(typeof(int), "x"),
            Expression.Constant((int) 5, typeof(int))
        ),
        new ParameterExpression[] { CS$0$0000 });

     return;
}
```

Compiler bietet Zugriff auf den Syntax Tree zur Laufzeit

AST in DLR


Sprachspezifisch
Source
Parser
Sprach-
spezifischer
AST
Transform
DLR AST
Transform
DLR

ExpressionTrees in C#

2010

Inheritance Hierarchy

System.Object
System.Linq.Expressions.Expression
System.Linq.Expressions.BinaryExpression
System.Linq.Expressions.BlockExpression
System.Linq.Expressions.ConditionalExpression
System.Linq.Expressions.ConstantExpression
System.Linq.Expressions.DebugInfoExpression
System.Linq.Expressions.DefaultExpression
System.Linq.Expressions.DynamicExpression
System.Linq.Expressions.GotoExpression
System.Linq.Expressions.IndexExpression
System.Linq.Expressions.InvocationExpression
System.Linq.Expressions.LabelExpression
System.Linq.Expressions.LambdaExpression
System.Linq.Expressions.ListInitExpression
System.Linq.Expressions.LoopExpression
System.Linq.Expressions.MemberExpression
System.Linq.Expressions.MemberInitExpression
System.Linq.Expressions.MethodCallExpression
System.Linq.Expressions.NewArrayExpression
System.Linq.Expressions.NewExpression
System.Linq.Expressions.ParameterExpression
System.Linq.Expressions.RuntimeVariablesExpression
System.Linq.Expressions.SwitchExpression
System.Linq.Expressions.TryExpression
System.Linq.Expressions.TypeBinaryExpression
System.Linq.Expressions.UnaryExpression

2008

Inheritance Hierarchy

System.Object
System.Linq.Expressions.Expression
System.Linq.Expressions.BinaryExpression
System.Linq.Expressions.ConditionalExpression
System.Linq.Expressions.ConstantExpression
System.Linq.Expressions.InvocationExpression
System.Linq.Expressions.LambdaExpression
System.Linq.Expressions.ListInitExpression
System.Linq.Expressions.MemberExpression
System.Linq.Expressions.MemberInitExpression
System.Linq.Expressions.MethodCallExpression
System.Linq.Expressions.NewArrayExpression
System.Linq.Expressions.NewExpression
System.Linq.Expressions.ParameterExpression
System.Linq.Expressions.TypeBinaryExpression
System.Linq.Expressions.UnaryExpression

Übersetzung in Microsoft Expression Trees

FORMELSPRACHE IN ACTION

Formelsprache in Action

Generieren des Expression Trees

```
var param = Expression.Parameter(typeof(int), "x");
Expression<Func<int, int>> ex2 =
    Expression.Lambda<Func<int, int>>(
        Expression.MakeBinary(
            ExpressionType.Subtract,
            param,
            Expression.Constant(1)),
        param);
```

Formelsprache in Action

Compilieren des Expression Trees

```
private static Func<Project, object> CompileFormula(
    ExpressionLanguageCommonTree ast)
{
    var parameter = Expression.Parameter(typeof(Project), "p");
    var expression = ProjectList.ConvertToExpressionTree(ast, parameter);
    var lambda = Expression.Lambda<Func<Project, object>>(
        Expression.Convert(expression, typeof(object)),
        parameter);

    return lambda.Compile();
}
```

Formelsprache in Action

Ausführen des kompilierten Expression Tree

```
public IEnumerable Projects
{
    get
    {
        return
            from p in Project.DemoData
            select new
            {
                p.ProjectName,
                p.Budget,
                p.TotalCost,
                CustomCol = this.CustomColumnFunction!=null
                    ? this.CustomColumnFunction(p) : null,
                BackgroundColor = this.CustomColumnFunction!=null
                    ? this.BackgroundFunction(p) : null
            };
    }
}
```

Ressourcen zum Thema ANTLR

- ANTLR
  - http://www.antlr.org/
  - Buch „The Definitive ANTLR Reference“ auf Amazon

Fragen?

R.STROPEK@CUBIDO.AT