Checkmarx CxEnterprise CxQL API DOM information

V8.4.0

November, 2016

Contents

1 CxDOM Types

The built-in types in CxDOM are listed here:

Types	Types (cont.)	Types(Cont.)
AccessorDecl	DestructorDecl	ParamDecl
ArgumentRef	EnumDecl	PostfixExpr
ArrayCreateExpr	EnumMemberDecl	PrimitiveExpr
ArrayElementRef	EventDecl	PropertyDecl
ArrayInitializer	EventRef	PropertyRef
AssemblyReference	Expression	PropertySetValueRef
AssignExpr	ExprStmt	RankSpecifier
AttachDelegateStmt	FieldDecl	RealLiteral
BaseRef	FieldRef	Reference
BinaryExpr	ForEachStmt	RemoveDelegateStmt
BooleanLiteral	GotoStmt	ReturnStmt
BreakStmt	IfStmt	Statement
BuiltInType	Import	StaticRef
Case	IndexerDecl	StringLiteral
CastExpr	IndexerRef	StructDecl
Catch	IntegerLiteral	SubExpr
CharLiteral	InterfaceDecl	SwitchStmt
CheckedStmt	IterationStmt	TernaryExpr
ClassDecl	LabeledStmt	ThisRef
Comment	LinePragma	ThrowStmt
CommentStmt	LocalRef	TryCatchFinallyStmt
CompileUnit	LockStmt	TypeDecl
ConstantDecl	MemberAccess	TypeOfExpr
ConstantDeclStmt	MemberDecl	TypeRef
ConstructorDecl	MethodDecl	UnaryExpr
ContinueStmt	MethodInvokeExpr	UncheckedStmt
CreateDelegateExpr	MethodRef	UnknownReference
CSharpGraph	NamespaceDecl	UsingStmt

Types	Types (cont.)	Types(Cont.)
CustomAttribute	NullLiteral	VariableDecl
Declarator	ObjectCreateExpr	VariableDeclStmt
DelegateDecl	OperatorDecl	
DelegateInvokeExpr	Param	

Example

In order to better understand each of these types, try the following query:

```
CxQL

result = All.FindByType(typeof(IfStmt));
Change "IfStmt" to one of the above types.
```

2 CSharpGraph methods

In the previous sections, we explained how to manipulate CxList, which are basically collections of basic code elements. Those basic elements are represented as instances of the CSharpGraph class or a class which inherits from CSharpGraph.

In some situations, it may be useful to manipulate directely basic code elements. It will give you the ability to implement home-made fine-grained queries in an optimized way.

This section describes the inheritance hierarchy of the CSharpGraph and its decendants.

Own methods:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1 MemberDecl : CSharpGraph

Own methods:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.1 TypeDecl : MemberDecl

Own methods:

- `public abstract int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public Checkmarx.Dom.TypeRefCollection BaseTypes { set; get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public Checkmarx.Dom.TypeRefCollection Generics { set; get; }`
- `public bool HasGenerics { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public Checkmarx.Dom.MemberDeclCollection Members { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeModifiers TypeAttributes { set; get; }`
- `public abstract Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`

- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.1.1 ClassDecl : TypeDecl

Own methods:

- `public override int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public Checkmarx.Dom.ClassDecl BaseClass { get; }`
- `public SymbolTable.OverloadableDefinition Constructors { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public override Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from TypeDecl:

- `public abstract int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public Checkmarx.Dom.TypeRefCollection BaseTypes { set; get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public Checkmarx.Dom.TypeRefCollection Generics { set; get; }`
- `public bool HasGenerics { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public Checkmarx.Dom.MemberDeclCollection Members { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeModifiers TypeAttributes { set; get; }`
- `public abstract Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`

- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.1.2 DelegateDecl : TypeDecl

Own methods:

- `public override int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.ParamDeclCollection Parameters { get; }`
- `public Checkmarx.Dom.TypeRef ReturnType { set; get; }`
- `public override Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from TypeDecl:

- `public abstract int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public Checkmarx.Dom.TypeRefCollection BaseTypes { set; get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public Checkmarx.Dom.TypeRefCollection Generics { set; get; }`
- `public bool HasGenerics { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public Checkmarx.Dom.MemberDeclCollection Members { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeModifiers TypeAttributes { set; get; }`
- `public abstract Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`

- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.1.1.3 EnumDecl : TypeDecl

Own methods:

- public override int DistanceToType(SymbolTable.IDefinition baseType, int distance)
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public new SymbolTable.Scope Scope { set; get; }
- public override Checkmarx.Dom.TypeKind TypeKind { get; }

Methods from TypeDecl:

- public abstract int DistanceToType(SymbolTable.IDefinition baseType, int distance)
- public Checkmarx.Dom.TypeRefCollection BaseTypes { set; get; }
- public SymbolTable.IDefinition Definition { get; }
- public Checkmarx.Dom.TypeRefCollection Generics { set; get; }
- public bool HasGenerics { get; }

- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public Checkmarx.Dom.MemberDeclCollection Members { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeModifiers TypeAttributes { set; get; }`
- `public abstract Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.1.4 InterfaceDecl : TypeDecl

Own methods:

- `public override int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public override Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from TypeDecl:

- `public abstract int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public Checkmarx.Dom.TypeRefCollection BaseTypes { set; get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public Checkmarx.Dom.TypeRefCollection Generics { set; get; }`
- `public bool HasGenerics { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public Checkmarx.Dom.MemberDeclCollection Members { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeModifiers TypeAttributes { set; get; }`
- `public abstract Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`

- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.1.5 StructDecl : TypeDecl

Own methods:

- `public override int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public SymbolTable.OverloadableDefinition Constructors { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public override Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from TypeDecl:

- `public abstract int DistanceToType(SymbolTable.IDefinition baseType, int distance)`
- `public Checkmarx.Dom.TypeRefCollection BaseTypes { set; get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public Checkmarx.Dom.TypeRefCollection Generics { set; get; }`
- `public bool HasGenerics { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public Checkmarx.Dom.MemberDeclCollection Members { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeModifiers TypeAttributes { set; get; }`
- `public abstract Checkmarx.Dom.TypeKind TypeKind { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`

- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.2 AccessorDecl : MemberDecl

Own methods:

- `public Checkmarx.Dom.AccessorModifiers AccessorModifier { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public Checkmarx.Dom.StatementCollection Statements { get; }`
- `public SymbolTable.Definition ValueDefinition { set; get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.1.3 ConstantDecl : MemberDecl

Own methods:

- public Checkmarx.Dom.DeclaratorCollection Declarators { get; }
- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public override Checkmarx.Dom.MemberKind MemberKind { get; }
- public override string Text { get; }
- public Checkmarx.Dom.TypeRef Type { set; get; }

Methods from MemberDecl:

- public bool IsCommentsExists()
- public bool IsCustomAttributesExists()
- public Checkmarx.Dom.Modifiers Attributes { set; get; }
- public Checkmarx.Dom.CommentStmtCollection Comments { get; }
- public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }
- public abstract Checkmarx.Dom.MemberKind MemberKind { get; }
- public override string Text { get; }

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.4 ConstructorDecl : MemberDecl

Own methods:

- `public SymbolTable.Definition BaseDefinition { set; get; }`
- `public Checkmarx.Dom.ParamCollection BaseParameters { get; }`
- `public SymbolTable.Definition ChainDefinition { set; get; }`
- `public Checkmarx.Dom.ParamCollection ChainParameters { get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string HashText { get; }`
- `public bool InvokeBase { set; get; }`
- `public bool InvokeChain { set; get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public string Name { set; get; }`
- `public Checkmarx.Dom.ParamDeclCollection Parameters { get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public Checkmarx.Dom.StatementCollection Statements { get; }`

- `public override string Text { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.5 DestructorDecl : MemberDecl

Own methods:

- `public SymbolTable.IDefinition Definition { get; }`

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public string Name { set; get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public Checkmarx.Dom.StatementCollection Statements { get; }`
- `public override string Text { get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.6 EnumMemberDecl : MemberDecl

Own methods:

- `public SymbolTable.IDefinition Definition { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.Expression Value { set; get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`

- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.7 EventDecl : MemberDecl

Own methods:

- `public Checkmarx.Dom.AccessorDecl AddAccessor { set; get; }`
- `public Checkmarx.Dom.DeclaratorCollection Declarators { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public Checkmarx.Dom.AccessorDecl RemoveAccessor { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeRef Type { set; get; }`
- `public bool UsesAccessors { set; get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`

- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.8 FieldDecl : MemberDecl

Own methods:

- `public Checkmarx.Dom.DeclaratorCollection Declarators { get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeRef Type { set; get; }`

Methods from MemberDecl:

- `public bool IsCommentsExists()`
- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CommentStmtCollection Comments { get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }`
- `public abstract Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`

- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.1.9 IndexerDecl : MemberDecl

Own methods:

- public SymbolTable.IDefinition Definition { get; }
- public Checkmarx.Dom.AccessorDecl GetAccessor { set; get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public bool HasGet { set; get; }
- public string HashText { get; }
- public bool HasSet { set; get; }
- public Checkmarx.Dom.TypeRef InterfaceType { set; get; }
- public override Checkmarx.Dom.MemberKind MemberKind { get; }
- public string Name { set; get; }
- public Checkmarx.Dom.ParamDeclCollection Parameters { get; }
- public new SymbolTable.Scope Scope { set; get; }
- public Checkmarx.Dom.AccessorDecl SetAccessor { set; get; }
- public Checkmarx.Dom.TypeRef Type { set; get; }

Methods from MemberDecl:

- public bool IsCommentsExists()
- public bool IsCustomAttributesExists()
- public Checkmarx.Dom.Modifiers Attributes { set; get; }
- public Checkmarx.Dom.CommentStmtCollection Comments { get; }
- public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }
- public abstract Checkmarx.Dom.MemberKind MemberKind { get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }

- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.1.10 MethodDecl : MemberDecl

Own methods:

- public bool IsReturnTypeCustomAttributesExists()
- public SymbolTable.IDefinition Definition { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public string HashText { get; }
- public bool HasTypeParameters { get; }
- public override Checkmarx.Dom.MemberKind MemberKind { get; }
- public string Name { set; get; }
- public Checkmarx.Dom.ParamDeclCollection Parameters { get; }
- public Checkmarx.Dom.TypeRef ReturnType { set; get; }
- public Checkmarx.Dom.CustomAttributeCollection ReturnTypeCustomAttributes { get; }
- public new SymbolTable.Scope Scope { set; get; }
- public Checkmarx.Dom.StatementCollection Statements { get; }
- public override string Text { get; }
- public Checkmarx.Dom.TypeRefCollection TypeParameters { set; get; }

Methods from MemberDecl:

- public bool IsCommentsExists()
- public bool IsCustomAttributesExists()
- public Checkmarx.Dom.Modifiers Attributes { set; get; }
- public Checkmarx.Dom.CommentStmtCollection Comments { get; }
- public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { get; }
- public abstract Checkmarx.Dom.MemberKind MemberKind { get; }
- public override string Text { get; }

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.1.11 OperatorDecl : MemberDecl

Own methods:

- `public static Checkmarx.Dom.OverloadableOperator OperatorFromString(string sop)`
- `public static string StringFromOperator(Checkmarx.Dom.OverloadableOperator sop)`
- `public string ConversionName { set; get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public Checkmarx.Dom.ParamDecl FirstParameter { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string HashText { get; }`
- `public override Checkmarx.Dom.MemberKind MemberKind { get; }`
- `public string Name { get; }`
- `public Checkmarx.Dom.OverloadableOperator Operator { set; get; }`
- `public Checkmarx.Dom.ParamDeclCollection Parameters { get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public Checkmarx.Dom.ParamDecl SecondParameter { set; get; }`
- `public Checkmarx.Dom.StatementCollection Statements { get; }`

- public override string **Text** { get; }
- public Checkmarx.Dom.TypeRef **Type** { set; get; }

Methods from MemberDecl:

- public bool **IsCommentsExists**()
- public bool **IsCustomAttributesExists**()
- public Checkmarx.Dom.Modifiers **Attributes** { set; get; }
- public Checkmarx.Dom.CommentStmtCollection **Comments** { get; }
- public Checkmarx.Dom.CustomAttributeCollection **CustomAttributes** { get; }
- public abstract Checkmarx.Dom.MemberKind **MemberKind** { get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.1.12 PropertyDecl : MemberDecl

Own methods:

- public SymbolTable.IDefinition **Definition** { get; }
- public Checkmarx.Dom.AccessorDecl **GetAccessor** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public bool **HasGet** { set; get; }
- public bool **HasSet** { set; get; }
- public override Checkmarx.Dom.MemberKind **MemberKind** { get; }
- public string **Name** { set; get; }
- public Checkmarx.Dom.AccessorDecl **SetAccessor** { set; get; }
- public override string **Text** { get; }
- public Checkmarx.Dom.TypeRef **Type** { set; get; }

Methods from MemberDecl:

- public bool **IsCommentsExists**()
- public bool **IsCustomAttributesExists**()
- public Checkmarx.Dom.Modifiers **Attributes** { set; get; }
- public Checkmarx.Dom.CommentStmtCollection **Comments** { get; }
- public Checkmarx.Dom.CustomAttributeCollection **CustomAttributes** { get; }
- public abstract Checkmarx.Dom.MemberKind **MemberKind** { get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father

- `public int p_no`

2.2 Expression : CSharpGraph

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1 Reference : Expression

Own methods:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`

- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.1 ArgumentRef : Reference

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string ParameterName { set; get; }`
- `public override string Text { get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.2 ArrayElementRef : Reference

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.ExpressionCollection Indices { get; }`
- `public Checkmarx.Dom.Expression Target { set; get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.3 BaseRef : Reference

Own methods:

- public override SymbolTable.IDefinition **Definition** { get; }
- public override string **FullName** { get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public override string **Text** { get; }

Methods from Reference:

- public Checkmarx.Dom.IDeclaration **DeclaredType** { set; get; }
- public virtual SymbolTable.IDefinition **Definition** { set; get; }
- public override string **Text** { get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father

- `public int p_no`

2.2.1.4 BuiltInType : Reference

Own methods:

- `public static bool isBuiltinType(string fullName)`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`
- `public override string TypeName { set; get; }`
- `public static int falseCount`
- `public static int trueCount`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`

- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.1.5 EventRef : Reference

Own methods:

- public string **EventName** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Target** { set; get; }

Methods from Reference:

- public Checkmarx.Dom.IDeclaration **DeclaredType** { set; get; }
- public virtual SymbolTable.IDefinition **Definition** { set; get; }
- public override string **Text** { get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }

- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.6 FieldRef : Reference

Own methods:

- `public string FieldName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Target { set; get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`

- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.1.7 IndexerRef : Reference

Own methods:

- public override SymbolTable.IDefinition Definition { get; }
- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.ExpressionCollection Indices { get; }
- public Checkmarx.Dom.Expression Target { set; get; }

Methods from Reference:

- public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }
- public virtual SymbolTable.IDefinition Definition { set; get; }
- public override string Text { get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }

- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.8 LocalRef : Reference

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`
- `public string VariableName { set; get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`

- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.1.9 MemberAccess : Reference

Own methods:

- public Checkmarx.Dom.CSharpGraph Declarator { get; }
- public Checkmarx.Dom.TypeRef DeclaratorType { get; }
- public string Def { get; }
- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public string MemberName { set; get; }
- public Checkmarx.Dom.Expression Target { set; get; }
- public override string Text { get; }

Methods from Reference:

- public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }
- public virtual SymbolTable.IDefinition Definition { set; get; }
- public override string Text { get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.10 MethodRef : Reference

Own methods:

- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string MethodName { set; get; }`
- `public Checkmarx.Dom.Expression Target { set; get; }`
- `public override string Text { get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.1.11 PropertyRef : Reference

Own methods:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public string **PropertyName** { set; get; }
- public Checkmarx.Dom.Expression **Target** { set; get; }

Methods from Reference:

- public Checkmarx.Dom.IDeclaration **DeclaredType** { set; get; }
- public virtual SymbolTable.IDefinition **Definition** { set; get; }
- public override string **Text** { get; }

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.12 PropertySetValueRef : Reference

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`

- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.13 ThisRef : Reference

Own methods:

- `public override SymbolTable.IDefinition Definition { get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.1.14 UnknownReference : Reference

Own methods:

- `public Checkmarx.Dom.CSharpGraph Declarator { get; }`
- `public Checkmarx.Dom.TypeRef DeclaratorType { get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`
- `public string VariableName { set; get; }`

Methods from Reference:

- `public Checkmarx.Dom.IDeclaration DeclaredType { set; get; }`
- `public virtual SymbolTable.IDefinition Definition { set; get; }`
- `public override string Text { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`

- `public int p_no`

2.2.2 PrimitiveExpr : Expression

Own methods:

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.2.1 BooleanLiteral : PrimitiveExpr

Own methods:

- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public override string Text { get; }
- public bool Value { set; get; }

Methods from PrimitiveExpr:

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.2.2 CharLiteral : PrimitiveExpr

Own methods:

- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public override string Text { get; }
- public char Value { set; get; }

Methods from PrimitiveExpr:

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.2.3 IntegerLiteral : PrimitiveExpr

Own methods:

- public string ExtendedType { set; get; }
- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public override string Text { get; }
- public long Value { set; get; }

Methods from PrimitiveExpr:

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.2.4 NullLiteral : PrimitiveExpr

Own methods:

- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public override string Text { get; }
- public string Value { set; get; }

Methods from PrimitiveExpr:

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.2.5 RealLiteral : PrimitiveExpr

Own methods:

- public override string FullName { get; }
- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public override string Text { get; }
- public double Value { set; get; }

Methods from PrimitiveExpr:

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.2.6 StringLiteral : PrimitiveExpr

Own methods:

- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`
- `public string Value { set; get; }`

Methods from PrimitiveExpr:

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.3 ArrayCreateExpr : Expression

Own methods:

- public Checkmarx.Dom.TypeRef **CreateType** { set; get; }
- public int **DimensionCount** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.ArrayInitializer **Initializer** { set; get; }
- public Checkmarx.Dom.RankSpecifierCollection **RankSpecifiers** { get; }
- public Checkmarx.Dom.ExpressionCollection **Sizes** { get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.4 ArrayInitializer : Expression

Own methods:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.ExpressionCollection **InitialValues** { get; }
- public override string **Text** { get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.5 AssignExpr : Expression

Own methods:

- public static Checkmarx.Dom.AssignOperator **OperatorFromString**(string op)
- public static string **StringFromOperator**(Checkmarx.Dom.AssignOperator op)

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Left** { set; get; }
- public Checkmarx.Dom.AssignOperator **Operator** { set; get; }
- public Checkmarx.Dom.Expression **Right** { set; get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.6 BinaryExpr : Expression

Own methods:

- public static Checkmarx.Dom.BinaryOperator **OperatorFromString**(string txt)

- `public static string StringFromOperator(Checkmarx.Dom.BinaryOperator bopEnum)`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Left { set; get; }`
- `public Checkmarx.Dom.BinaryOperator Operator { set; get; }`
- `public Checkmarx.Dom.Expression Right { set; get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.7 CastExpr : Expression

Own methods:

- `public Checkmarx.Dom.Expression Expression { set; get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public new Checkmarx.Dom.IDeclaration ResultType { get; }`
- `public Checkmarx.Dom.TypeRef TargetType { set; get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.8 CreateDelegateExpr : Expression

Own methods:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public string **MethodName** { set; get; }
- public Checkmarx.Dom.Expression **Target** { set; get; }
- public Checkmarx.Dom.TypeRef **Type** { set; get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigV alue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.9 DelegateInvokeExpr : Expression

Own methods:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }

- public Checkmarx.Dom.ParamCollection **Parameters** { get; }
- public Checkmarx.Dom.Expression **Target** { set; get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Member** { set; get; }
- public SymbolTable.IDefinition **ResultType** { set; get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.10 MethodInvokeExpr : Expression

Own methods:

- public void **computeDefinition**()
- public SymbolTable.IDefinition **Definition** { get; }
- public override string **FullName** { set; get; }

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.ParamCollection Parameters { get; }
- public Checkmarx.Dom.MethodRef Target { set; get; }
- public override string Text { get; }

Methods from Expression:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.Expression Member { set; get; }
- public SymbolTable.IDefinition ResultType { set; get; }
- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.2.11 ObjectCreateExpr : Expression

Own methods:

- public Checkmarx.Dom.MemberDeclCollection AnonymousBody { set; get; }

- `public SymbolTable.IDefinition ConstructorDefinition { set; get; }`
- `public Checkmarx.Dom.TypeRef CreateType { set; get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.ParamCollection Parameters { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.12 PostfixExpr : Expression

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Left { set; get; }`
- `public Checkmarx.Dom.PostfixOperator Operator { set; get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.13 SubExpr : Expression

Own methods:

- `public Checkmarx.Dom.Expression Expression { set; get; }`
- `public override string FullName { get; }`

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.14 TernaryExpr : Expression

Own methods:

- `public Checkmarx.Dom.Expression False { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Test { set; get; }`
- `public Checkmarx.Dom.Expression True { set; get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.15 TypeOfExpr : Expression

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.TypeRef Type { set; get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.2.16 UnaryExpr : Expression

Own methods:

- `public static Checkmarx.Dom.UnaryOperator OperatorFromString(string txt)`
- `public static string StringFromOperator(Checkmarx.Dom.UnaryOperator uOpEnum)`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.UnaryOperator Operator { set; get; }`
- `public Checkmarx.Dom.Expression Right { set; get; }`

Methods from Expression:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Member { set; get; }`
- `public SymbolTable.IDefinition ResultType { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3 Statement : CSharpGraph

Own methods:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.1 AttachDelegateStmt : Statement

Own methods:

- `public Checkmarx.Dom.TypeRef Event { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression Listener { set; get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.2 BreakStmt : Statement

Own methods:

- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`

- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)
- public virtual bool isSubTypeOf(string subTypeName)
- public int BlockId { set; get; }
- public System.Collections.Generic.HashSet<int> BlockIdArr { get; }
- public Checkmarx.Dom.TypeDecl Class { get; }
- public int DOMDepth { get; }
- public virtual Checkmarx.Dom.IGraph _father { set; get; }
- public string Father { get; }
- public virtual string FullName { set; get; }
- public virtual Checkmarx.Dom.GraphTypes GraphType { get; }
- public int language { set; get; }
- public Checkmarx.Dom.LinePragma LinePragma { set; get; }
- public Checkmarx.Dom.MemberDecl Method { get; }
- public Checkmarx.Dom.NamespaceDecl Namespace { get; }
- public virtual int _no { set; get; }
- public string Scope { get; }
- public virtual string ShortName { get; }
- public abstract string Text { get; }
- public string TokenPos { get; }
- public virtual string TypeName { set; get; }
- public System.Collections.IDictionary UserData { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.3 CheckedStmt : Statement

Own methods:

- public override Checkmarx.Dom.GraphTypes GraphType { get; }
- public Checkmarx.Dom.StatementCollection Statements { get; }

Methods from Statement:

- public override string Text { get; }

Methods from CSharpGraph:

- public void AddUserData(string key, object data)
- public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)
- public static System.Type GetTypeFromString(string s)
- public object GetUserData(string key)
- public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)

- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.4 CommentStmt : Statement

Own methods:

- `public Checkmarx.Dom.Comment Comment { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`

- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.5 ConstantDeclStmt : Statement

Own methods:

- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.DeclaratorCollection Declarators { get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.TypeRef Type { set; get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`

- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.6 ContinueStmt : Statement

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`

- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.7 ExprStmt : Statement

Own methods:

- `public Checkmarx.Dom.Expression Expression { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`

- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.8 ForEachStmt : Statement

Own methods:

- public Checkmarx.Dom.Expression **Collection** { set; get; }
- public override string **FullName** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.IterationType **IterationType** { get; }
- public string **Name** { set; get; }
- public Checkmarx.Dom.StatementCollection **Statements** { get; }
- public Checkmarx.Dom.TypeRef **Type** { set; get; }
- public Checkmarx.Dom.VariableDeclStmt **VariableDeclarator** { set; get; }

Methods from Statement:

- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved

- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.9 GotoStmt : Statement

Own methods:

- public Checkmarx.Dom.Expression **CaseLabel** { set; get; }
- public override string **FullName** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public string **Label** { set; get; }

Methods from Statement:

- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigV alue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.10 IfStmt : Statement

Own methods:

- `public Checkmarx.Dom.Expression Condition { set; get; }`
- `public Checkmarx.Dom.StatementCollection FalseStatements { get; }`
- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.StatementCollection TrueStatements { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.11 IterationStmt : Statement

Own methods:

- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.StatementCollection Increment { get; }`
- `public Checkmarx.Dom.StatementCollection Init { get; }`
- `public Checkmarx.Dom.IterationType IterationType { set; get; }`
- `public Checkmarx.Dom.StatementCollection Statements { get; }`
- `public Checkmarx.Dom.Expression Test { set; get; }`
- `public bool TestFirst { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.12 LabeledStmt : Statement

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string Label { set; get; }`
- `public Checkmarx.Dom.Statement Statement { set; get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.13 LockStmt : Statement

Own methods:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.StatementCollection **Statements** { get; }
- public Checkmarx.Dom.Expression **Target** { set; get; }

Methods from Statement:

- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.14 RemoveDelegateStmt : Statement

Own methods:

- public Checkmarx.Dom.TypeRef **Event** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **Listener** { set; get; }

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.15 ReturnStmt : Statement

Own methods:

- `public Checkmarx.Dom.Expression Expression { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.16 SwitchStmt : Statement

Own methods:

- `public Checkmarx.Dom.CaseCollection Cases { get; }`
- `public Checkmarx.Dom.Expression Condition { set; get; }`
- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`

- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.17 ThrowStmt : Statement

Own methods:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.Expression **ToThrow** { set; get; }

Methods from Statement:

- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigV
alue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }

- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.18 TryCatchFinallyStmt : Statement

Own methods:

- public Checkmarx.Dom.CatchCollection **CatchClauses** { set; get; }
- public Checkmarx.Dom.StatementCollection **Finally** { get; }
- public override string **FullName** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.StatementCollection **Try** { get; }

Methods from Statement:

- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigV alue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }

- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.19 UncheckedStmt : Statement

Own methods:

- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.StatementCollection **Statements** { get; }

Methods from Statement:

- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }

- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.3.20 UsingStmt : Statement

Own methods:

- public Checkmarx.Dom.VariableDeclStmt **Declaration** { set; get; }
- public bool **DeclaresResource** { get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.StatementCollection **Statements** { get; }
- public Checkmarx.Dom.Expression **Target** { set; get; }

Methods from Statement:

- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigV alue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }

- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.3.21 VariableDeclStmt : Statement

Own methods:

- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.DeclaratorCollection Declarators { get; }`
- `public int DimensionCount { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.TypeRef Type { set; get; }`

Methods from Statement:

- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`

- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.4 AssemblyReference : CSharpGraph

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string HintPath { set; get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.5 Case : CSharpGraph

Own methods:

- `public Checkmarx.Dom.Expression Condition { set; get; }`
- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public bool IsDefault { set; get; }`
- `public Checkmarx.Dom.StatementCollection Statements { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.6 Catch : CSharpGraph

Own methods:

- public Checkmarx.Dom.TypeRef **CatchExceptionType** { set; get; }
- public SymbolTable.IDefinition **Definition** { get; }
- public override string **FullName** { set; get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public string **LocalName** { set; get; }
- public Checkmarx.Dom.StatementCollection **Statements** { get; }
- public override string **Text** { get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.7 Comment : CSharpGraph

Own methods:

- `public string CommentText { set; get; }`
- `public bool DocComment { set; get; }`
- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.8 CompileUnit : CSharpGraph

Own methods:

- `public CompileUnit()`
- `public bool IsCustomAttributesExists()`
- `public void resetLanguage()`
- `public void resetLanguage(bool correctNSLanguage)`
- `public Checkmarx.Dom.CustomAttributeCollection AssemblyCustomAttributes { get; }`
- `public System.Collections.ArrayList FileName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.ImportCollection Imports { get; }`
- `public System.DateTime Modified { set; get; }`
- `public System.Collections.ArrayList Name { get; }`
- `public Checkmarx.Dom.NamespaceDeclCollection Namespaces { get; }`
- `public Checkmarx.Dom.AssemblyReferenceCollection ReferencedAssemblies { get; }`
- `public new SymbolTable.Scope Scope { set; get; }`
- `public SymbolTable.ScopeCollection ScopeTable { get; }`
- `public override string Text { get; }`
- `public bool IsAttributed`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`

- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.9 CustomAttribute : CSharpGraph

Own methods:

- `public static string StringFromAttributeTarget(Checkmarx.Dom.AttributeTarget at)`
- `public Checkmarx.Dom.AttributeTarget AttributeTarget { set; get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string Name { set; get; }`
- `public Checkmarx.Dom.ExpressionCollection Parameters { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.10 Declarator : CSharpGraph

Own methods:

- `public SymbolTable.IDefinition Definition { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public Checkmarx.Dom.Expression InitExpression { set; get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.11 Import : CSharpGraph

Own methods:

- `public string Alias { set; get; }`
- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public new string Namespace { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.12 NamespaceDecl : CSharpGraph

Own methods:

- public SymbolTable.IDefinition **Definition** { get; }
- public override Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public Checkmarx.Dom.ImportCollection **Imports** { get; }
- public string **Name** { set; get; }
- public Checkmarx.Dom.NamespaceDeclCollection **Namespaces** { get; }
- public new SymbolTable.Scope **Scope** { set; get; }
- public override string **Text** { get; }
- public Checkmarx.Dom.TypeDeclCollection **Types** { set; get; }

Methods from CSharpGraph:

- public void **AddUserData**(string key, object data)
- public void **AddUserData**(string key, object data, string boolConfigKey,bool defaultConfigValue)
- public Checkmarx.Dom.CSharpGraph **GetAncOfType**(System.Type objType)
- public static System.Type **GetTypeFromString**(string s)
- public object **GetUserData**(string key)
- public bool **IsAncOf**(Checkmarx.Dom.CSharpGraph child)
- public virtual bool **isSubTypeOf**(string subTypeName)
- public int **BlockId** { set; get; }
- public System.Collections.Generic.HashSet<int> **BlockIdArr** { get; }
- public Checkmarx.Dom.TypeDecl **Class** { get; }
- public int **DOMDepth** { get; }
- public virtual Checkmarx.Dom.IGraph **_father** { set; get; }
- public string **Father** { get; }
- public virtual string **FullName** { set; get; }
- public virtual Checkmarx.Dom.GraphTypes **GraphType** { get; }
- public int **language** { set; get; }
- public Checkmarx.Dom.LinePragma **LinePragma** { set; get; }
- public Checkmarx.Dom.MemberDecl **Method** { get; }
- public Checkmarx.Dom.NamespaceDecl **Namespace** { get; }
- public virtual int **_no** { set; get; }
- public string **Scope** { get; }
- public virtual string **ShortName** { get; }
- public abstract string **Text** { get; }
- public string **TokenPos** { get; }
- public virtual string **TypeName** { set; get; }
- public System.Collections.IDictionary **UserData** { set; get; }
- public System.Collections.Generic.List Children
- public static bool FullnameResolved
- public static int nodeId
- public int NodeId
- public Checkmarx.Dom.IGraph p_father
- public int p_no

2.13 Param : CSharpGraph

Own methods:

- `public Checkmarx.Dom.ParamDirection Direction { set; get; }`
- `public override string FullName { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.Expression Value { set; get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.14 ParamDecl : CSharpGraph

Own methods:

- `public bool IsCustomAttributesExists()`
- `public Checkmarx.Dom.Modifiers Attributes { set; get; }`
- `public Checkmarx.Dom.CustomAttributeCollection CustomAttributes { set; get; }`
- `public SymbolTable.IDefinition Definition { get; }`
- `public Checkmarx.Dom.ParamDirection Direction { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public bool IsOptional { set; get; }`
- `public bool IsParams { set; get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`
- `public Checkmarx.Dom.TypeRef Type { set; get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigV alue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.15 Project : CSharpGraph

Own methods:

- `public Project()`
- `public Checkmarx.Dom.CompileUnitCollection CompileUnits { get; }`
- `public string FileName { set; get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string Name { set; get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.16 RankSpecifier : CSharpGraph

Own methods:

- `public int Dimensions { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.17 Solution : CSharpGraph

Own methods:

- `public int getNumberOfMethods()`
- `public Checkmarx.Dom.Solution Merge(Checkmarx.Dom.Solution s)`
- `public Solution()`
- `public static Checkmarx.Dom.Solution operator +(Checkmarx.Dom.Solution s1, Checkmarx.Dom.Solution s2)`
- `public string FileName { set; get; }`
- `public override string FullName { get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public string Name { set; get; }`
- `public Checkmarx.Dom.ProjectsCollection Projects { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.18 TypeRef : CSharpGraph

Own methods:

- `public bool IsRankSpecifierCollectionExists()`
- `public Checkmarx.Dom.TypeRef ArrayElementType { set; get; }`
- `public Checkmarx.Dom.RankSpecifierCollection ArrayRanks { set; get; }`
- `public SymbolTable.IDefinition Definition { set; get; }`
- `public string ExtendedTypeName { set; get; }`
- `public Checkmarx.Dom.TypeRefCollection Generics { set; get; }`
- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public bool HasGenerics { get; }`
- `public override string Text { get; }`
- `public override string TypeName { set; get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

2.19 VariableDecl : CSharpGraph

Own methods:

- `public override Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public override string Text { get; }`

Methods from CSharpGraph:

- `public void AddUserData(string key, object data)`
- `public void AddUserData(string key, object data, string boolConfigKey,bool defaultConfigValue)`
- `public Checkmarx.Dom.CSharpGraph GetAncOfType(System.Type objType)`
- `public static System.Type GetTypeFromString(string s)`
- `public object GetUserData(string key)`
- `public bool IsAncOf(Checkmarx.Dom.CSharpGraph child)`
- `public virtual bool isSubTypeOf(string subTypeName)`
- `public int BlockId { set; get; }`
- `public System.Collections.Generic.HashSet<int> BlockIdArr { get; }`
- `public Checkmarx.Dom.TypeDecl Class { get; }`
- `public int DOMDepth { get; }`
- `public virtual Checkmarx.Dom.IGraph _father { set; get; }`
- `public string Father { get; }`
- `public virtual string FullName { set; get; }`
- `public virtual Checkmarx.Dom.GraphTypes GraphType { get; }`
- `public int language { set; get; }`
- `public Checkmarx.Dom.LinePragma LinePragma { set; get; }`
- `public Checkmarx.Dom.MemberDecl Method { get; }`
- `public Checkmarx.Dom.NamespaceDecl Namespace { get; }`
- `public virtual int _no { set; get; }`
- `public string Scope { get; }`
- `public virtual string ShortName { get; }`
- `public abstract string Text { get; }`
- `public string TokenPos { get; }`
- `public virtual string TypeName { set; get; }`
- `public System.Collections.IDictionary UserData { set; get; }`
- `public System.Collections.Generic.List Children`
- `public static bool FullnameResolved`
- `public static int nodeId`
- `public int NodeId`
- `public Checkmarx.Dom.IGraph p_father`
- `public int p_no`

3 CSharpGraph Examples

This section some common usages of the CSharpGraph API.

Precise Name Filtering

The methods CxList.FindByName and CxList.FindByShortName can be used for name filtering. However, if you need more flexibility and want to express a more precise filter, you can do this by looking at each code element, using the CSharpGraph API.

```
CxQL

 This example demonstrates the use of CSharpGraph.ShortName.
 The input source code is:

 public class sumClass{
      public int sumAll (int parameter, int parameterparameter){
             int hello = 0, hellohello = 1;
             int hellohellohello = 2, hellohellohellohello = 3;
             return parameter + hello + hellohellohellohello
                    + hellohellohello + hellohello + parameterparameter;
      }
 }

CxList vars = All.FindByType(typeof(UnknownReference));
foreach(CxList v in vars){
      // Get the CSharpGraph of the only code element in v
      CSharpGraph g = v.data.GetByIndex(0) as CSharpGraph;
      // Check a condition on the ShortName
      if(g != null && g.ShortName != null && g.ShortName.Length < 13){
             // Add only elements which meet the condition
             result.Add(v);
      }
}

 The result would consist of 3 items:
 - parameter
 - hello
 - hellohello
```

Filtering String Content

Another example of how to use CSharpGraph.ShortName.

```
CxQL

 This example demonstrates the use of CSharpGraph.ShortName.
 The input source code is:

 public class DbClass{
      public int myQueries (int query_id){
             switch(query_id){
             case 0: return "UPDATE table SET comment = ''";
             case 1: return "SELECT * FROM table WHERE id = 1";
             case 2: return "  insert into table set select = 'all'  ";
             case 3: return "  select name from table where id = 3  ";
             default: return "DELETE from table";
             }
      }
 }

CxList strings = All.FindByType(typeof(StringLiteral));
char[] trimChars = new char[6] {' ', '\t', '"', '(', '\r', '\n'};
CxList SQL = strings.FindByName("*select *", false);
foreach (CxList sql in SQL)
{
      // Get the CSharpGraph of the only code element in sql
      CSharpGraph gr = sql.data.GetByIndex(0) as CSharpGraph;
      string name = gr.ShortName.TrimStart(trimChars);
      // Check a condition on name
      if (name.ToLower().StartsWith("select"))
      {
             // Add only elements which meet the condition
             result.Add(sql);
      }
}

 The result would consist of 2 items:
 - case 1
 - case 3
```

Find Variables at the same Line

This example shows how to use CSharpGraph.LinePragma in order to find all the variables present at the same line as one of the element of a given CxList.

```
CxQL

 This example demonstrates the use of CSharpGraph.ShortName.
 The input source code is:

 public class myClass{
      public int myMethod (int a, int b){
             int x = a + b;
             int y = a * b;
             int z = x * x + b * b;
             return x + y + z;
      }
 }

CxList x = All.FindByShortName("x");
CxList variables = All.FindByType(typeof(UnknownReference));
foreach (KeyValuePair<int, IGraph> dic in x.data)
{
      if (dic.Value.LinePragma != null)
      {
        result.Add(variables.FindByPosition(dic.Value.LinePragma.Line));
        }
}

 The result would consist of 9 items:
 - 2 results at line 5
 - 4 retults at line 7
 - 3 results at line 8
```

4 IDeclaration and IDefinition

4.1 public interface IDeclaration

```
IDefinition Definition { get;}
GraphTypes GraphType { get;}
```

4.2 public interface IDefinition

```
int Id { get; set; }
string Name { get; set; }
string FullName { get; }
Scope Scope { get; set; }
IDeclaration SourceGraph { get; set; }
IDeclaration Type { get; set; }
```

4.3 public interface ISymbol : IDefinition

```
IDefinition DeclaredType { get; set; }
IDefinition ActualType { get; set; }
```

Signatures from IDefinition

```
int Id { get; set; }
string Name { get; set; }
string FullName { get; }
Scope Scope { get; set; }
IDeclaration SourceGraph { get; set; }
IDeclaration Type { get; set; }
```

4.4 public class Definition : IDefinition

```
ArrayList AddMemberDefinitionRange(ArrayList defId);
ArrayList AddMemberInstanceRange(ArrayList instId);
bool AddMemberRange(ArrayList defIds,ArrayList instIds);
int AddNewDefinitionID(int defId);
int AddNewInstanceID(int instId);
bool AddNewMember(int defId, int instId);
void computeFullName();
void ConvertToOldMembersDS();
ArrayList Definition_Id { get; set; }
string FullName { get; }
int GetHashCode();
int GetMemberInstByDef(int defId);
int Id { get; set; }
bool InsertMember(int pos, int defId, int instId);
ArrayList InsertMemberDefinitionRange(ArrayList defId);
ArrayList InsertMemberInstanceRange(ArrayList instId);
bool InsertMemberRange(.ArrayList defIds, .ArrayList instIds);
ArrayList Instance_Id { get; set; }
Definition.CxArrayList MembersDefinition_id { get; set; }
Definition.CxArrayList MembersInstance_id { get; set; }
MembersDefinition MembersList { get; set; }
string Name { get; set; }
Scope Scope { get; set; }
void SetFullName(string fullName);
IDeclaration SourceGraph { get; set; }
string ToString();
IDeclaration Type { get; set; }
```

4.5 public class OverloadableDefinition : Definition

```
public DefinitionCollection Definitions
```

Methods from Definition:

```
ArrayList AddMemberDefinitionRange(ArrayList defId);
ArrayList AddMemberInstanceRange(ArrayList instId);
bool AddMemberRange(ArrayList defIds,ArrayList instIds);
int AddNewDefinitionID(int defId);
int AddNewInstanceID(int instId);
bool AddNewMember(int defId, int instId);
void computeFullName();
void ConvertToOldMembersDS();
ArrayList Definition_Id { get; set; }
string FullName { get; }
int GetHashCode();
int GetMemberInstByDef(int defId);
int Id { get; set; }
bool InsertMember(int pos, int defId, int instId);
ArrayList InsertMemberDefinitionRange(ArrayList defId);
ArrayList InsertMemberInstanceRange(ArrayList instId);
bool InsertMemberRange(.ArrayList defIds, .ArrayList instIds);
ArrayList Instance_Id { get; set; }
Definition.CxArrayList MembersDefinition_id { get; set; }
Definition.CxArrayList MembersInstance_id { get; set; }
MembersDefinition MembersList { get; set; }
string Name { get; set; }
Scope Scope { get; set; }
void SetFullName(string fullName);
IDeclaration SourceGraph { get; set; }
string ToString();
IDeclaration Type { get; set; }
```