Type-Directed Completion of Partial Expressions

Daniel Perelman†

Sumit Gulwani‡ Thomas Ball‡ Dan Grossman†

†University of Washington

‡Microsoft Research Redmond

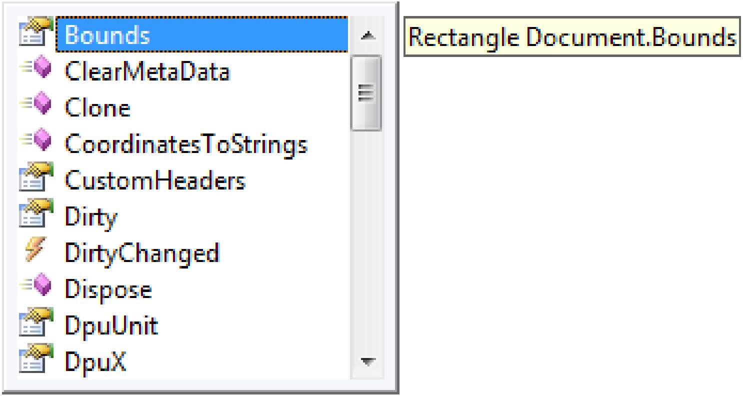
June 12, 2012

I want to shrink an image...

```
Document image = ...; Size newSize = ...;
image.Shrink(newSize)
```

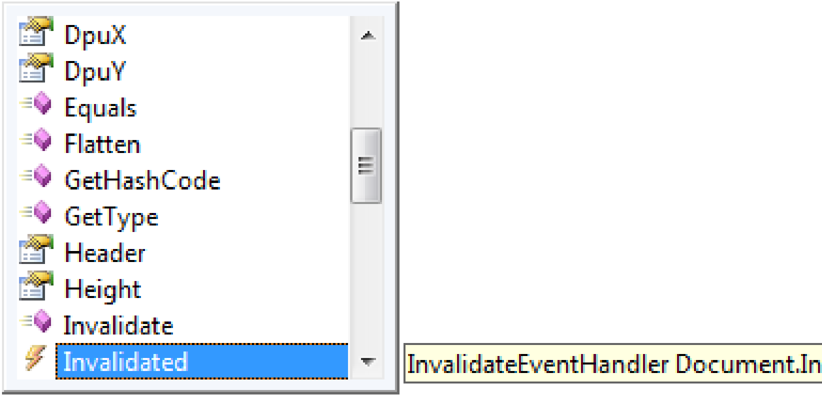
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
image.
```



I want to shrink an image...

```
Document image = ...; Size newSize = ...;
image.
```

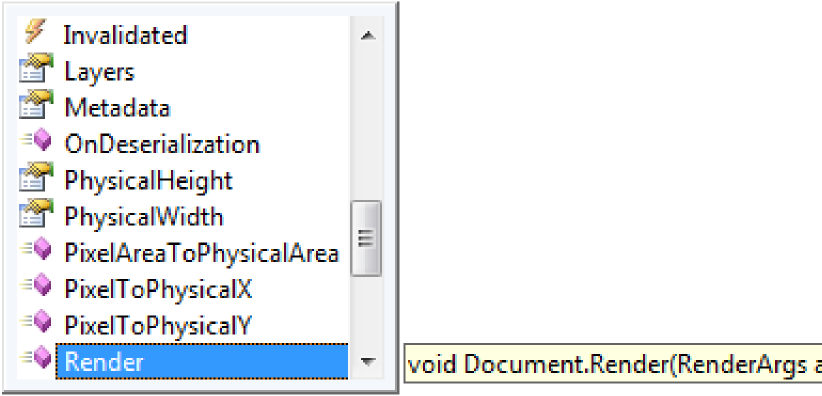


DpuX
DpuY
Equals
Flatten
GetHashCode
GetType
Header
Height
Invalidate
Invalidated

InvalidateEventHandler Document.In

I want to shrink an image...

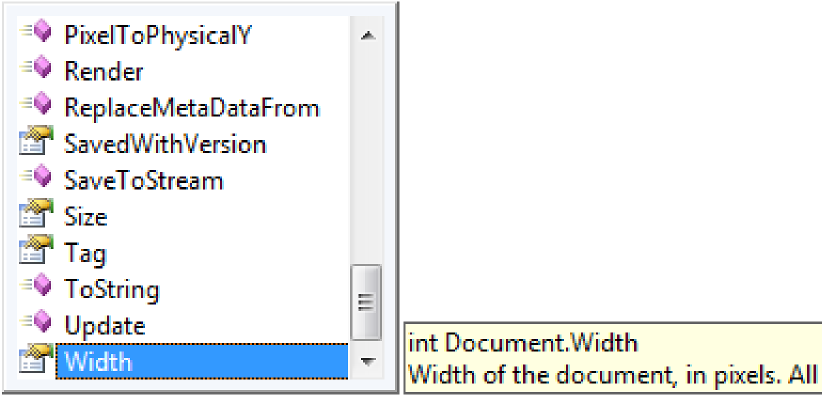
```
Document image = ...; Size newSize = ...;
image.
```



void Document.Render(RenderArgs a

I want to shrink an image...

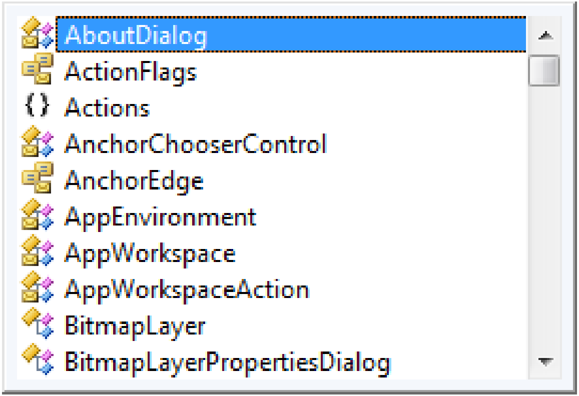
```
Document image = ...; Size newSize = ...;
image.
```



int Document.Width
Width of the document, in pixels. All

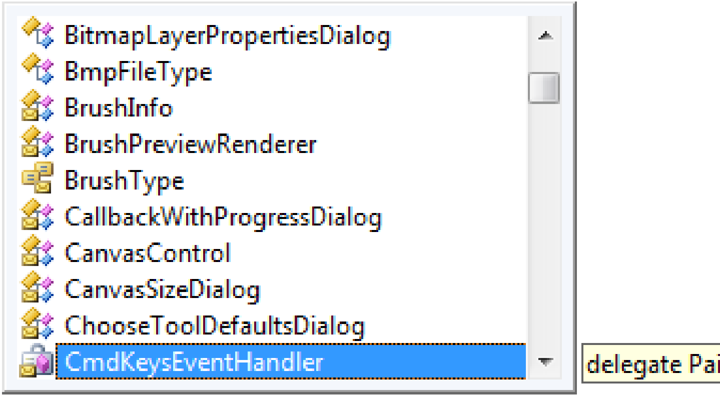
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.
```



I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.
```

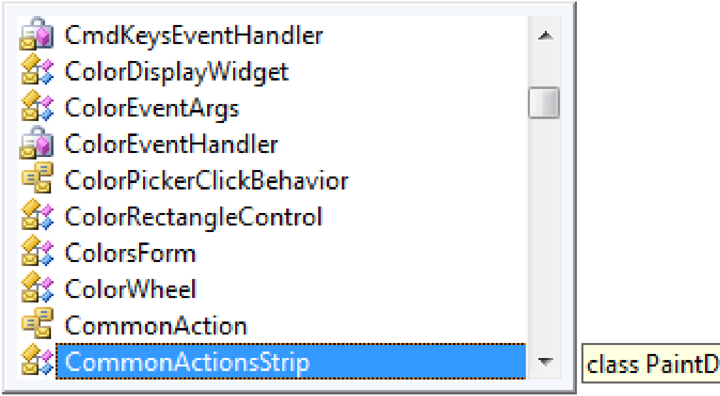


BitmapLayerPropertiesDialog
BmpFileType
BrushInfo
BrushPreviewRenderer
BrushType
CallbackWithProgressDialog
CanvasControl
CanvasSizeDialog
ChooseToolDefaultsDialog
CmdKeysEventHandler

delegate Pai

I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.
```

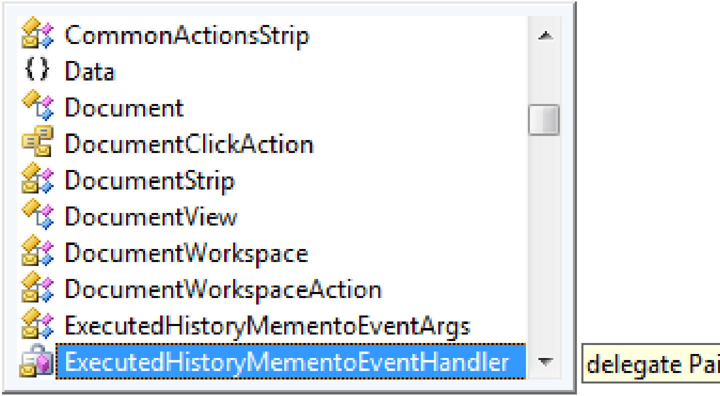


CmdKeysEventHandler
ColorDisplayWidget
ColorEventArgs
ColorEventHandler
ColorPickerClickBehavior
ColorRectangleControl
ColorsForm
ColorWheel
CommonAction
CommonActionsStrip

class PaintD

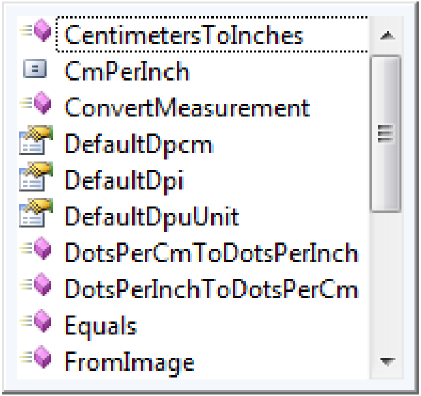
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.
```



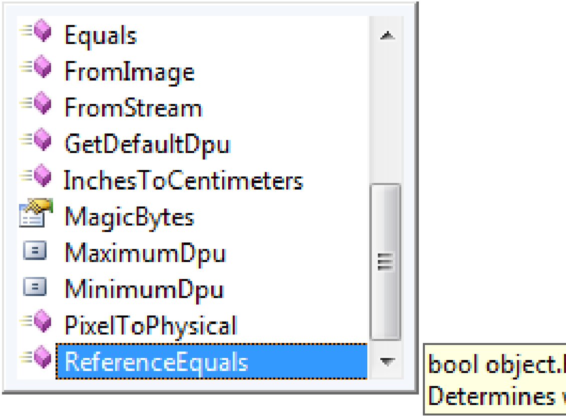
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Document.
```



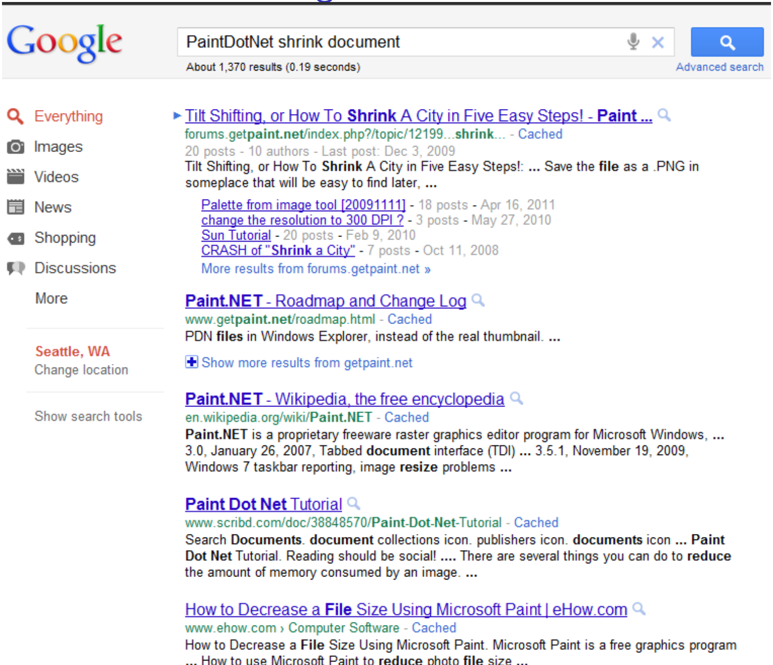
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Document.
```



bool object.I
Determines

I want to shrink an image...



About 1,370 results (0.19 seconds)

▸ Tilt Shifting, or How To **Shrink** A City in Five Easy Steps! - **Paint ...**
forums.get**paint.net**/index.php?/topic/12199...**shrink**... - Cached
20 posts - 10 authors - Last post: Dec 3, 2009
Tilt Shifting, or How To **Shrink** A City in Five Easy Steps!: **...** Save the **file** as a .PNG in someplace that will be easy to find later, **...**

Palette from image tool [20091111] - 18 posts - Apr 16, 2011
change the resolution to 300 DPI ? - 3 posts - May 27, 2010
Sun Tutorial - 20 posts - Feb 9, 2010
CRASH of "**Shrink** a City" - 7 posts - Oct 11, 2008
More results from forums.getpaint.net »

Paint.NET - Roadmap and Change Log
www.get**paint.net**/roadmap.html - Cached
PDN **files** in Windows Explorer, instead of the real thumbnail. **...**

⊞ Show more results from getpaint.net

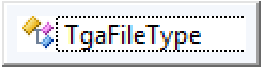
Paint.NET - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Paint.NET** - Cached
Paint.NET is a proprietary freeware raster graphics editor program for Microsoft Windows, **...** 3.0, January 26, 2007, Tabbed **document** interface (TDI) **...** 3.5.1, November 19, 2009, Windows 7 taskbar reporting, image **resize** problems **...**

Paint Dot Net Tutorial
www.scribd.com/doc/38848570/**Paint-Dot-Net**-Tutorial - Cached
Search **Documents**. **document** collections icon. publishers icon. **documents** icon **... Paint Dot Net** Tutorial. Reading should be social! **....** There are several things you can do to **reduce** the amount of memory consumed by an image. **...**

How to Decrease a **File** Size Using Microsoft Paint | eHow.com
www.ehow.com › Computer Software - Cached
How to Decrease a **File** Size Using Microsoft Paint. Microsoft Paint is a free graphics program **...** How to use Microsoft Paint to **reduce** photo **file** size **...**

I want to shrink an image...

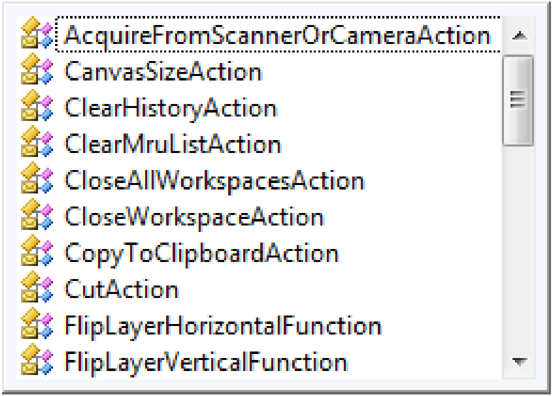
```
Document image = ...; Size newSize = ...;
PaintDotNet.Data.
```



TgaFileType

I want to shrink an image...

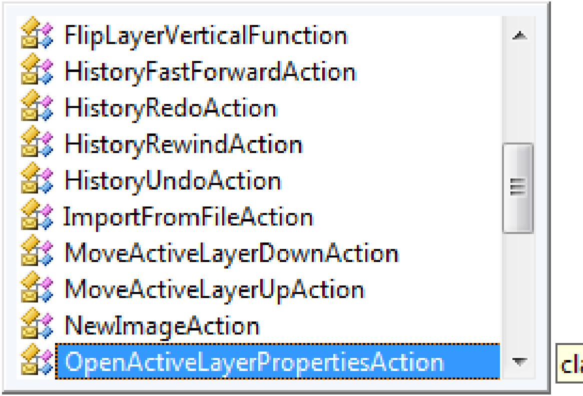
```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.
```



AcquireFromScannerOrCameraAction
CanvasSizeAction
ClearHistoryAction
ClearMruListAction
CloseAllWorkspacesAction
CloseWorkspaceAction
CopyToClipboardAction
CutAction
FlipLayerHorizontalFunction
FlipLayerVerticalFunction

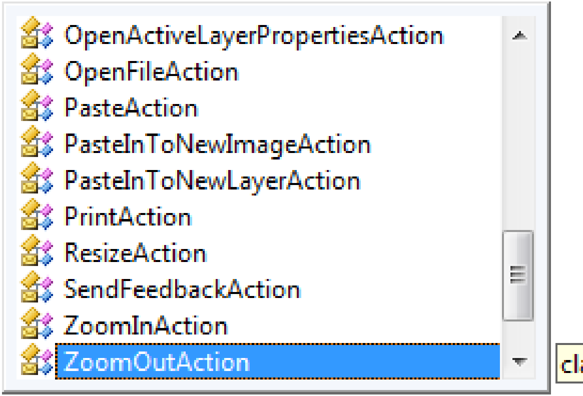
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.
```



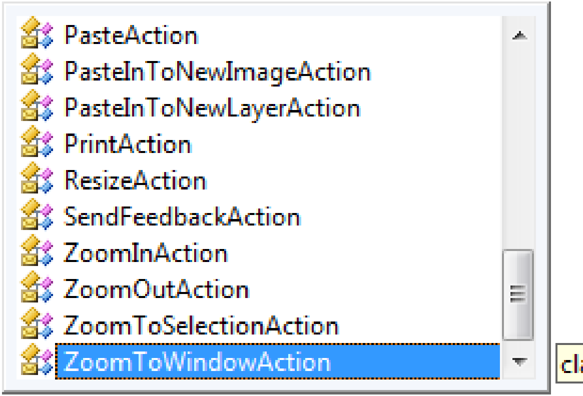
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.
```



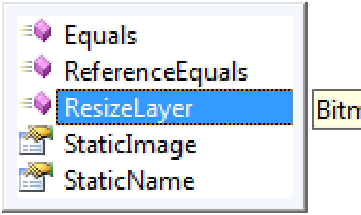
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.
```



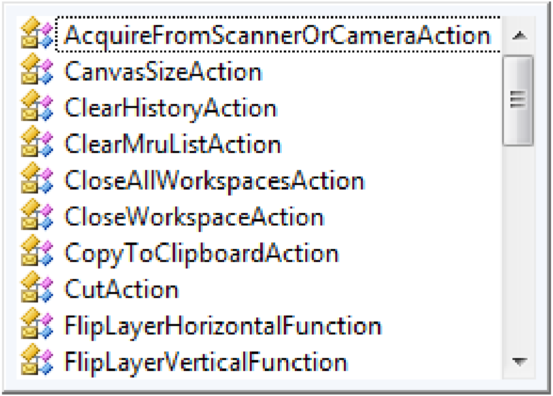
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.ResizeAction.
```



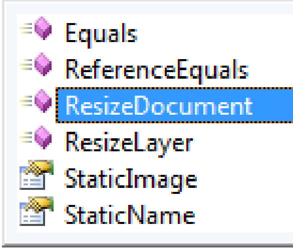
I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.
```



I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.CanvasSizeAction.
```



I want to shrink an image...

```
Document image = ...; Size newSize = ...;
PaintDotNet.Actions.CanvasSizeAction.
    .ResizeDocument(
    /* PaintDotNet.Document image */,
    /* System.Drawing.Size size */,
    /* PaintDotNet.AnchorEdge edge */,
    /* PointDotNet.ColorBgra bgColor */);
```

Programmer thought process

- I have a `Document` and a `Size`
- I want to shrink the `Document`
- There must be a method

Programmer thought process

- I have a `Document` and a `Size`
- I want to shrink the `Document`
- There must be a method

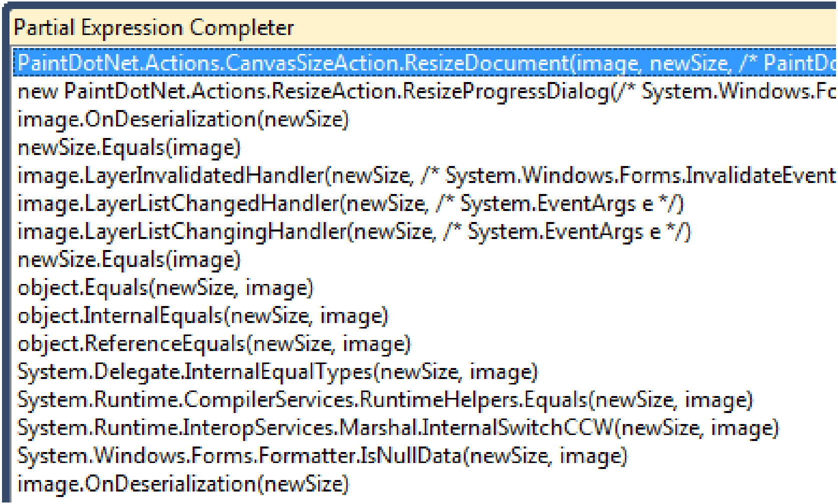
- Current code completion
  - Left-to-right
  - Complete, alphabetic list of just next token
  - Very limited filtering

Proposed workflow

```
Document image = ...; Size newSize = ...;
var newImage = ?({image, newSize})
```

Proposed workflow

```
Document image = ...; Size newSize = ...;
var newImage = ?({image, newSize})
```



Partial Expression Completer

```
PaintDotNet.Actions.CanvasSizeAction.ResizeDocument(image, newSize, /* PaintDo
new PaintDotNet.Actions.ResizeAction.ResizeProgressDialog(/* System.Windows.Fo
image.OnDeserialization(newSize)
newSize.Equals(image)
image.LayerInvalidatedHandler(newSize, /* System.Windows.Forms.InvalidateEvent
image.LayerListChangedHandler(newSize, /* System.EventArgs e */)
image.LayerListChangingHandler(newSize, /* System.EventArgs e */)
newSize.Equals(image)
object.Equals(newSize, image)
object.InternalEquals(newSize, image)
object.ReferenceEquals(newSize, image)
System.Delegate.InternalEqualTypes(newSize, image)
System.Runtime.CompilerServices.RuntimeHelpers.Equals(newSize, image)
System.Runtime.InteropServices.Marshal.InternalSwitchCCW(newSize, image)
System.Windows.Forms.Formatter.IsNullData(newSize, image)
image.OnDeserialization(newSize)
```

Programmer thought process

- I have a `Document` and a `Size`
- I want to shrink the `Document`
- There must be a method

- Query should contain what the programmer knows
  - Some values and types the expression should involve
  - Loose syntactic structure
- Query shouldn't require what the programmer doesn't know
  - Names
  - Argument order
  - Other arguments
- Show "best" results first
- Similar in spirit to Prospector [Mandelin et. al., PLDI'05]

Overview

- Expression of API queries as partial expressions
- Algorithm to generate results quickly in ranked order
- Experiment showing simple queries represent real code well

Unknown method queries

- Ex. `?({image, size})`
  - ⇒ `PaintDotNet.Actions.CanvasSizeAction`
    `.ResizeDocument(img, size, ◇, ◇)`
  - ⇒ `PaintDotNet.Functional.Func.Bind(◇, size, img)`
  - ⇒ `PaintDotNet.Pair.Create(size, img)`
  - ⇒ `PaintDotNet.Quadruple.Create(size, img, ◇, ◇)`
  - ⇒ `PaintDotNet.Triple.Create(size, img, ◇)`
  - ⇒ `PaintDotNet.PropertySystem`
    `.StaticListChoiceProperty`
    `.CreateForEnum(img, size, ◇)`
  - ⇒ `System.Drawing.Size.Equals(size, img)`
  - ⇒ `System.Object.ReferenceEquals(size, img)`

Unknown lookup queries

- Ex. `float f = pointPair.*`
  - ⇒ `pointPair.P1.X`
  - ⇒ `pointPair.P1.Y`
  - ⇒ `pointPair.P2.X`
  - ⇒ `pointPair.P2.Y`
  - ⇒ `pointPair.Midpoint.X`
  - ⇒ `pointPair.Midpoint.Y`
  - ⇒ `pointPair.FirstValidValue().X`
  - ⇒ `pointPair.Length`

Unknown expression queries

- Ex. `XmlReader xr = ?`
    - ⇒ `System.Xml.XmlReader.Create(◇)`
    - ⇒ `new System.Xml.XmlNodeReader(◇)`
    - ⇒ `System.Data.SqlTypes.SqlXml.Null.CreateReader()`
    - ⇒ `new System.Xml.XmlNodeReader(◇).ReadSubtree()`
    - ⇒ `new System.Xml.XmlValidatingReader(◇).Reader`
    - ⇒ `Microsoft.SqlServer.Server.SqlContext`
      `.TriggerContext.EventData.CreateReader()`
    - ⇒ `new System.Xml.XmlValidatingReader(◇)`
      `.Reader.ReadSubtree()`

Partial expression language

$$
\begin{array}{lrcl}
\text{(a)} & e & ::= & \mathit{call} \mid \mathit{varName} \mid e\,\texttt{.}\,\mathit{fieldName} \mid e\,\texttt{:=}\,e \mid e\,\texttt{<}\,e \\
 & \mathit{call} & ::= & \mathit{methodName}\texttt{(}e_1\texttt{,}\ldots\texttt{,}e_n\texttt{)} \\
\text{(b)} & \widetilde{e} & ::= & \widetilde{a} \mid \underset{\sim}{\mathbf{?}} \mid \diamond \\
 & \widetilde{a} & ::= & e \mid \widetilde{a}\,\texttt{.}\,\underset{\sim}{\mathbf{*}} \mid \widetilde{\mathit{call}} \mid \widetilde{e}\,\texttt{:=}\,\widetilde{e} \mid \widetilde{e}\,\texttt{<}\,\widetilde{e} \\
 & \widetilde{\mathit{call}} & ::= & \underset{\sim}{\mathbf{?}}\texttt{(}\{\widetilde{e}_1\texttt{,}\ldots\texttt{,}\widetilde{e}_n\}\texttt{)} \mid \mathit{methodName}\texttt{(}\widetilde{e}_1\texttt{,}\ldots\texttt{,}\widetilde{e}_n\texttt{)}
\end{array}
$$

Partial expression language

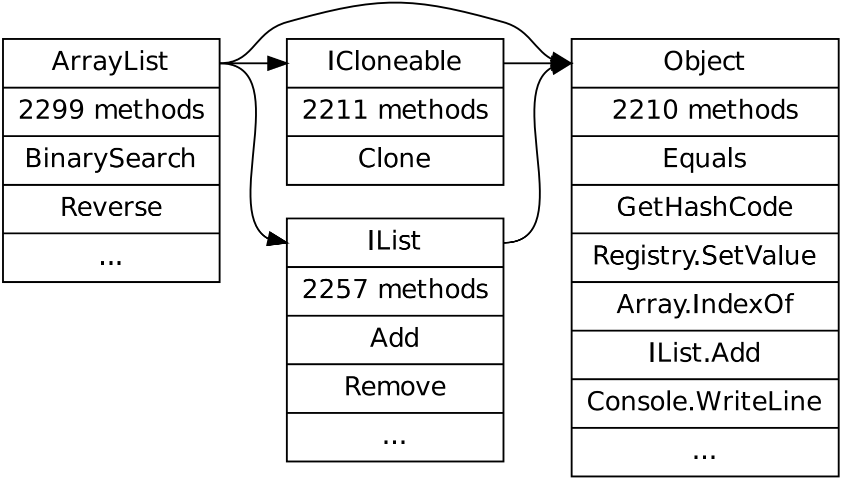
$$\begin{array}{llcl}
\text{(a)} & e & ::= & \mathit{call} \mid \mathit{varName} \mid e\,\mathtt{.}\,\mathit{fieldName} \mid e\,\mathtt{:=}\,e \mid e\,\mathtt{<}\,e \\
 & \mathit{call} & ::= & \mathit{methodName}\,\mathtt{(}e_1\,\mathtt{,}\,\ldots\,\mathtt{,}\,e_n\mathtt{)} \\
\text{(b)} & \widetilde{e} & ::= & \widetilde{a} \mid \utilde{\mathbf{?}} \mid \diamond \\
 & \widetilde{a} & ::= & e \mid \widetilde{a}\,\mathtt{.}\,\utilde{\mathbf{*}} \mid \widetilde{\mathit{call}} \mid \widetilde{e}\,\mathtt{:=}\,\widetilde{e} \mid \widetilde{e}\,\mathtt{<}\,\widetilde{e} \\
 & \widetilde{\mathit{call}} & ::= & \utilde{\mathbf{?}}\,\mathtt{(}\{\widetilde{e}_1\,\mathtt{,}\,\ldots\,\mathtt{,}\,\widetilde{e}_n\}\mathtt{)} \mid \mathit{methodName}\,\mathtt{(}\widetilde{e}_1\,\mathtt{,}\,\ldots\,\mathtt{,}\,\widetilde{e}_n\mathtt{)}
\end{array}$$

- Ex. $\utilde{\mathbf{?}}\mathtt{(}\{\mathtt{strBuilder.}\utilde{\mathbf{*}}\mathtt{,\ e.}\utilde{\mathbf{*}}\}\mathtt{)}$
  $\Rightarrow \utilde{\mathbf{?}}\mathtt{(}\{\mathtt{strBuilder,\ e.StackTrace}\}\mathtt{)}$
  $\Rightarrow \mathtt{strBuilder.Append(e.StackTrace)}$

Algorithm

- Problem: given query, generate completions

Method index by parameter type



ArrayList
2299 methods
BinarySearch
Reverse
...

ICloneable
2211 methods
Clone

IList
2257 methods
Add
Remove
...

Object
2210 methods
Equals
GetHashCode
Registry.SetValue
Array.IndexOf
IList.Add
Console.WriteLine
...

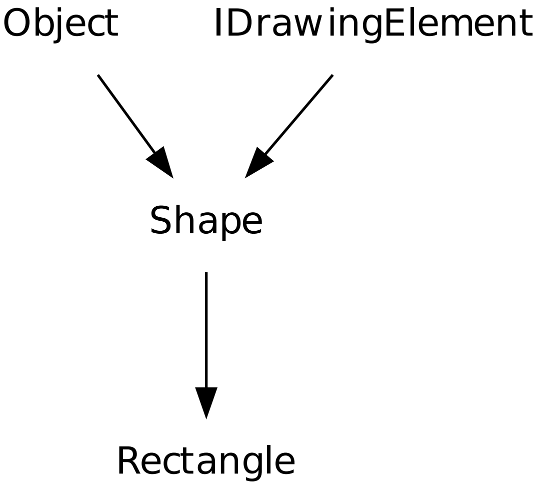
Infinite results

- Problem: too many results
  - inefficient to generate thousands of results to show only 20 to the programmer
  - programmer does not want to look at every result
  - result set is often infinite
- Ex. `var res = foo.*;`
  - ⇒ `foo`
  - ⇒ `foo.GetType()`
  - ⇒ `foo.GetType().GetType()`
  - ⇒ `foo.GetType().GetType().GetType()`
  - ⇒ `foo.GetType().GetType().GetType().GetType()`
  - ⇒ ...
- Solution: generate in ranked order

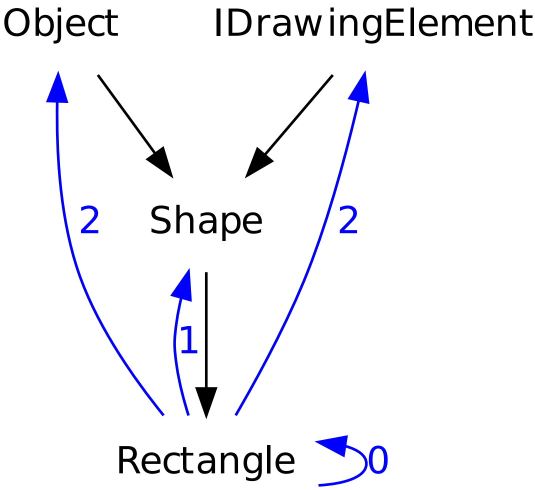
Algorithm

- Simple structually recursive algorithm
- Group by type to minimize redundant work
- Generate results in ranking order
    - Allows determination of top *n* without computing all results

Heuristics: Type distance



Heuristics: Type distance



Heuristics: Length

- Number of field/property lookups or method calls added

Heuristics: Length

- Number of field/property lookups or method calls added
- `?({strBuilder.*,e.*})`

  Good (1): ⇒ `strBuilder.Append(e.StackTrace)`

  Bad (3): ⇒ `strBuilder.Clear().Append(e.Data.Count)`

Heuristics: Inferred abstract types

Example usages elsewhere in codebase:

```
string f = Path.GetTempFileName(); ...;
File.Delete(f);
```

```
File.Delete(Path.Combine(dir, filename));
```

```
if(File.Exists(Path.Combine(otherDir, file))) {...}
```

Query:

```
string p = Path.GetTempFileName();
?({p})
⇒ GetCursor(p)
⇒ File.Delete(p)
⇒ File.Exists(p)
```

Ranking function

- Linear combination of these and other heuristics
- Sensitivity analysis showed these are most important and coefficients do not matter much

Outline

Motivation

Approach
- Language
- Algorithm
- Ranking

Experiment
- Results

Related work

Conclusion

Experiment

- Automated test of expressiveness of partial expressions
- Generated queries for each call and looked at rank of actual call in query results

- Advantage: able to do many queries
- Disadvantage: many of the method calls are not ones a programmer would need API discovery for

Experiment

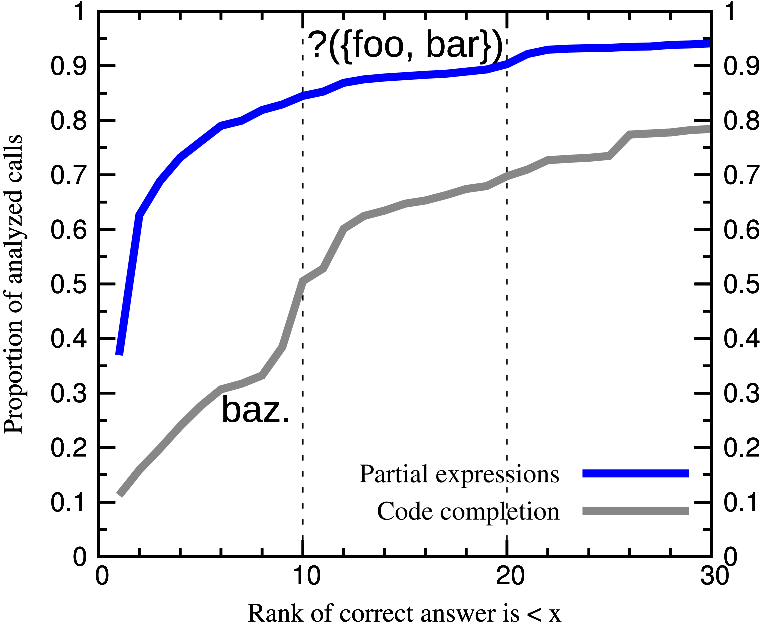
- Used Microsoft CCI to disassemble mature C# projects
- Converted every call with at least 3 arguments (including receiver) to a query with 1 or 2 arguments (including receiver)
  - For `ResizeDocument(document, size, anchorEdge, background)` 16 queries would be generated:
    $\Rightarrow$ $\underset{\sim}{\mathbf{?}}$`(document)`
    $\Rightarrow$ $\underset{\sim}{\mathbf{?}}$`(size)`
    $\Rightarrow$ $\underset{\sim}{\mathbf{?}}$`(anchorEdge)`
    $\Rightarrow$ $\underset{\sim}{\mathbf{?}}$`(background)`
    $\Rightarrow$ $\underset{\sim}{\mathbf{?}}$`(document, size)`
    $\Rightarrow$ $\underset{\sim}{\mathbf{?}}$`(document, background)`
    $\Rightarrow$ ...
- Report rank for best-performing query for each call

Projects used

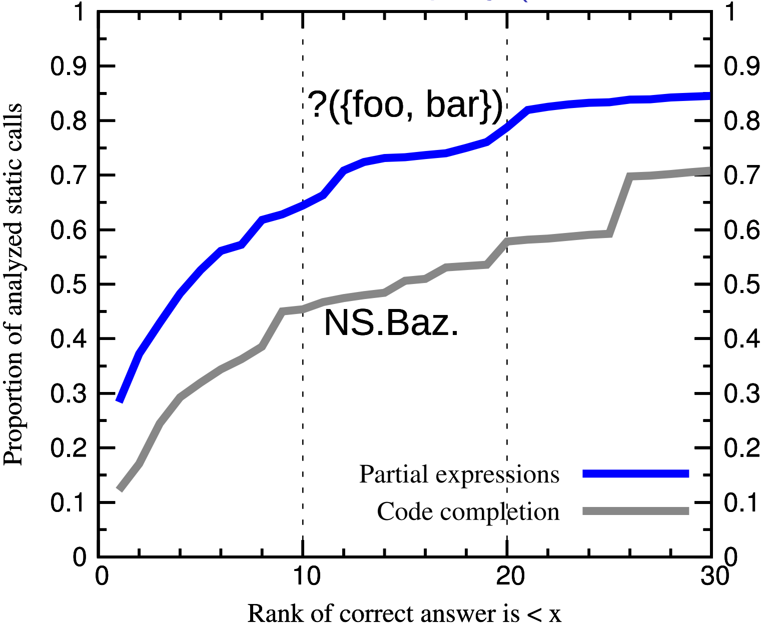
- Paint.NET image editor
- Windows Installer XML library
- Gnome Do program launcher
- Banshee music player
- .NET core libraries
- Family.Show (WPF example application)
- LiveGeometry geometry visualizer

- Scale: .NET contains 280,000 methods in 30,000 types
- Analyzed 21,176 method calls in these applications

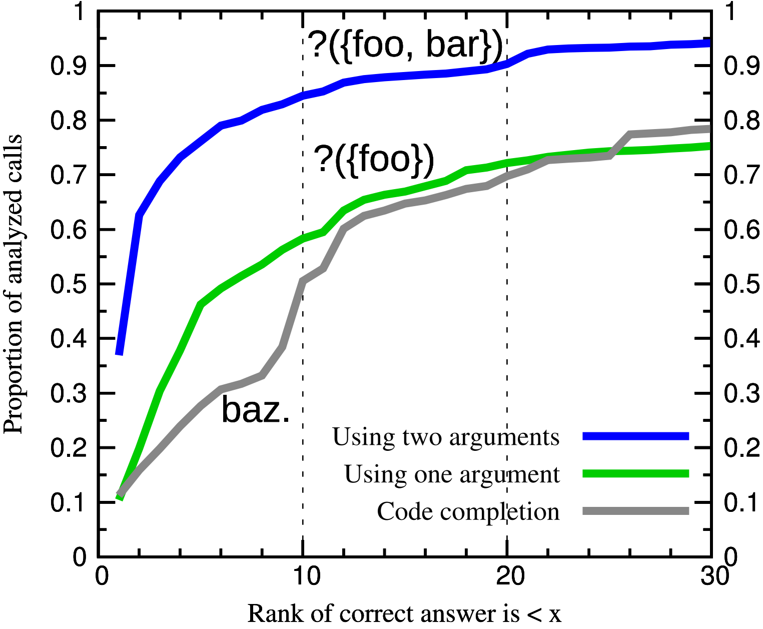
CDF of rank for best method query



CDF of rank for best method query (correct is static)



CDF of rank for best method query



Other experiments

- Time: unknown method queries take under 0.1 second
- Ran similar experiments on other partial expression templates
- Similar results: one argument or one lookup could be predicted within the top 10 about 80% of the time

Related work

- Lots of other work on API discovery discussed in paper

Related work

- Lots of other work on API discovery discussed in paper

- Prospector (for Java) [Mandelin et. al., PLDI'05]
  - Input is target type
    - Similar to `XmlReader xr = ?` query
  - Uses mined expressions which convert from one type to another
  - Output is chain of mined expressions starting with some local

  - Advantage: able to synthesize larger expressions
  - Disadvantage: queries only specify a single input type and a single output type

Contributions

- Expressed API searches in terms of partial expressions
- Leveraged rich type structure to reduce information needed for queries
- Automated experiments across large codebases show small partial expressions often match real method calls

- Created Visual Studio plugin
  - `https://pec.codeplex.com/`