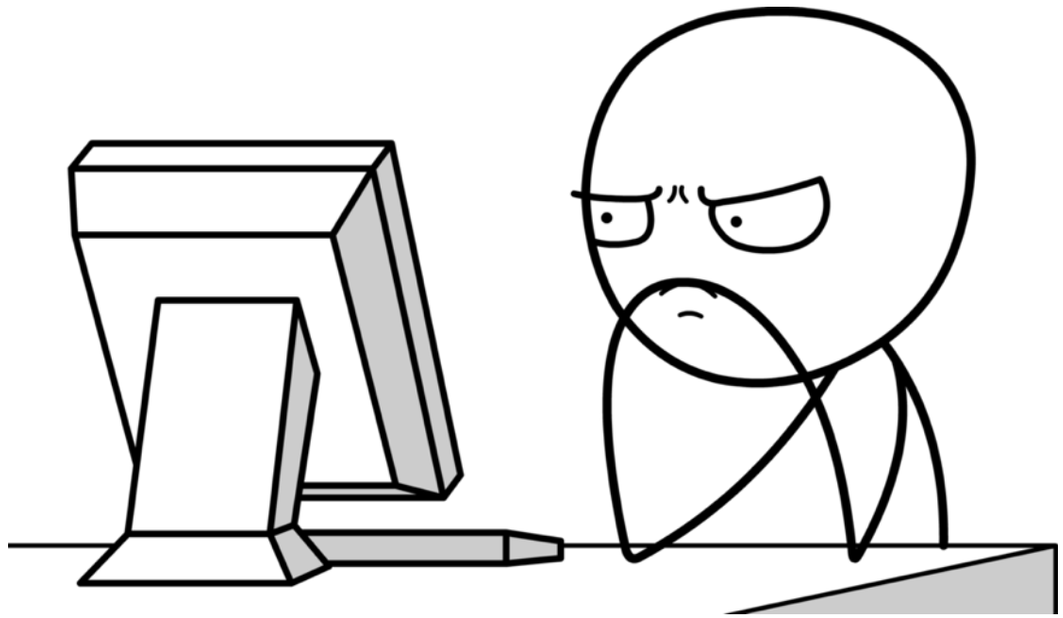
Making Sense of Online Code Snippets

Liangju Li

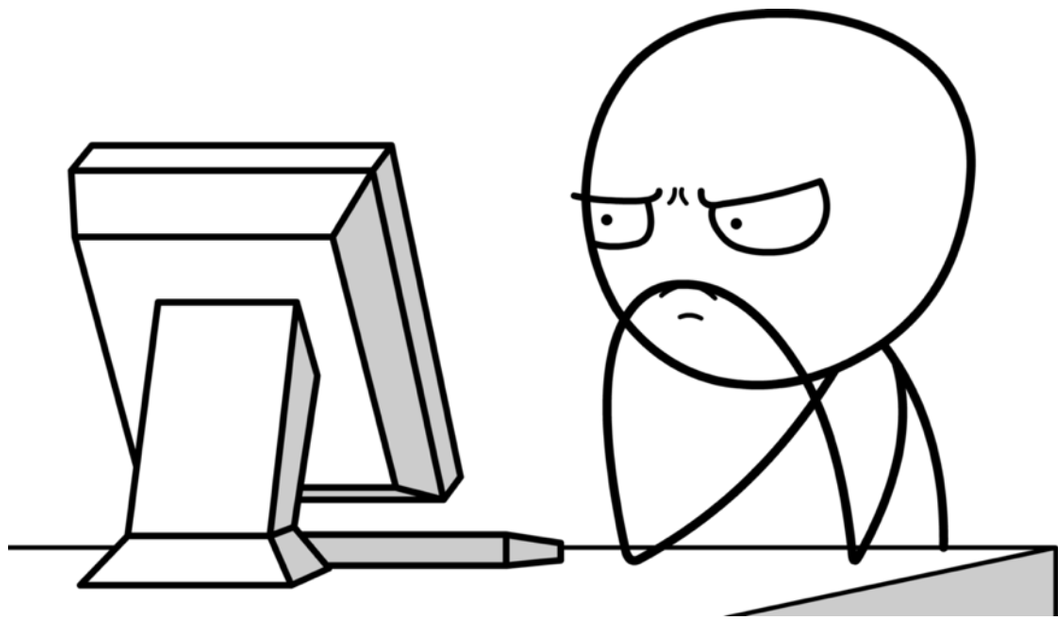
Motivation

During the SD and SE…



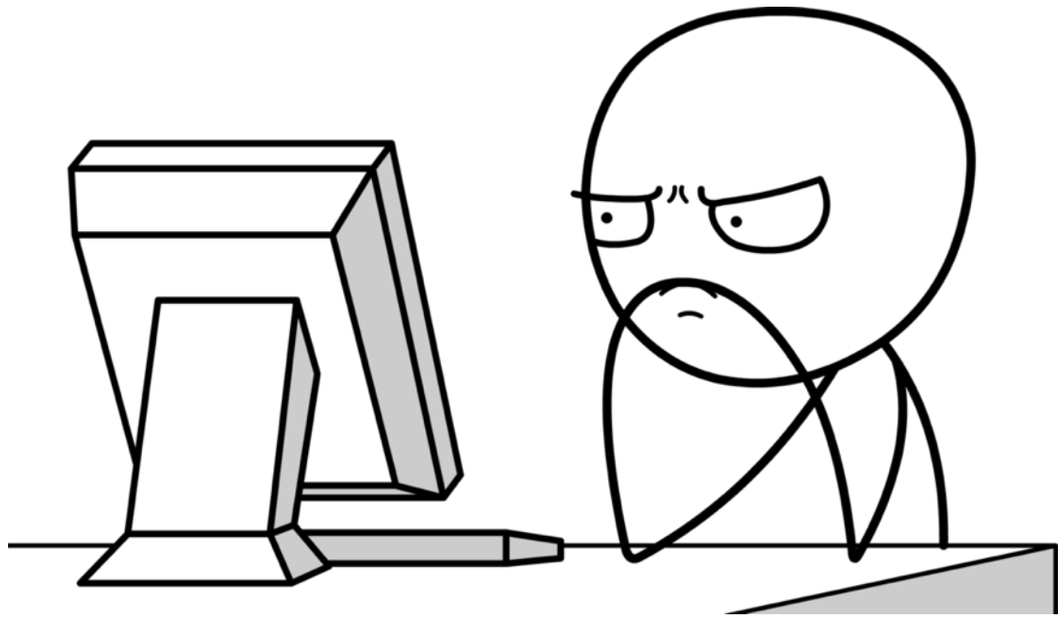
We might have some problems …

- Reuse existing libraries and frameworks
- Many frameworks are complex
- Lack of documentation or examples



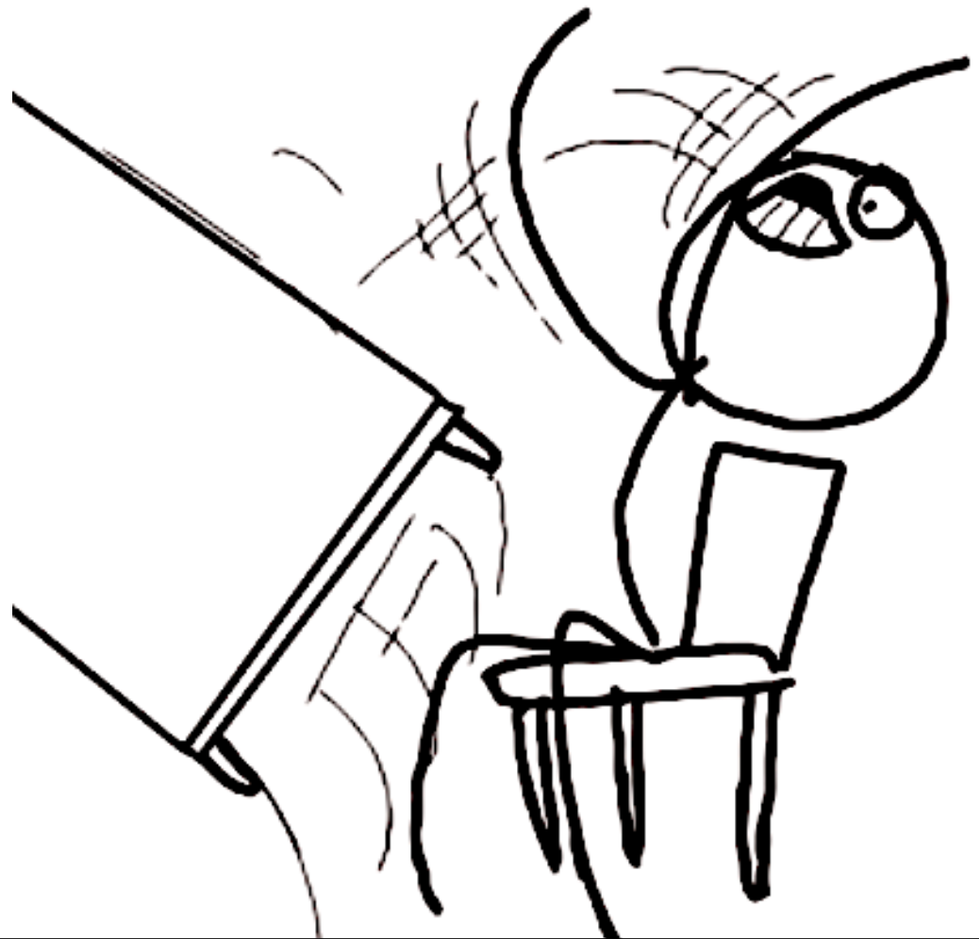
For example…
Google Play Service API for Android…

- Some documents are out of date.
- Few code examples.

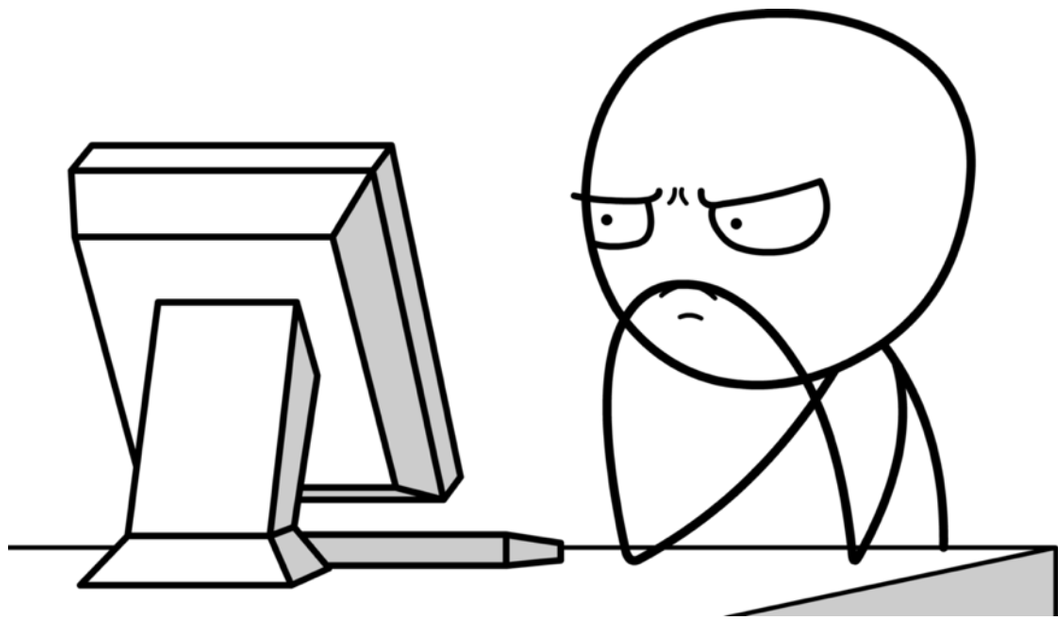


For example…
Google Play Service API for Android…

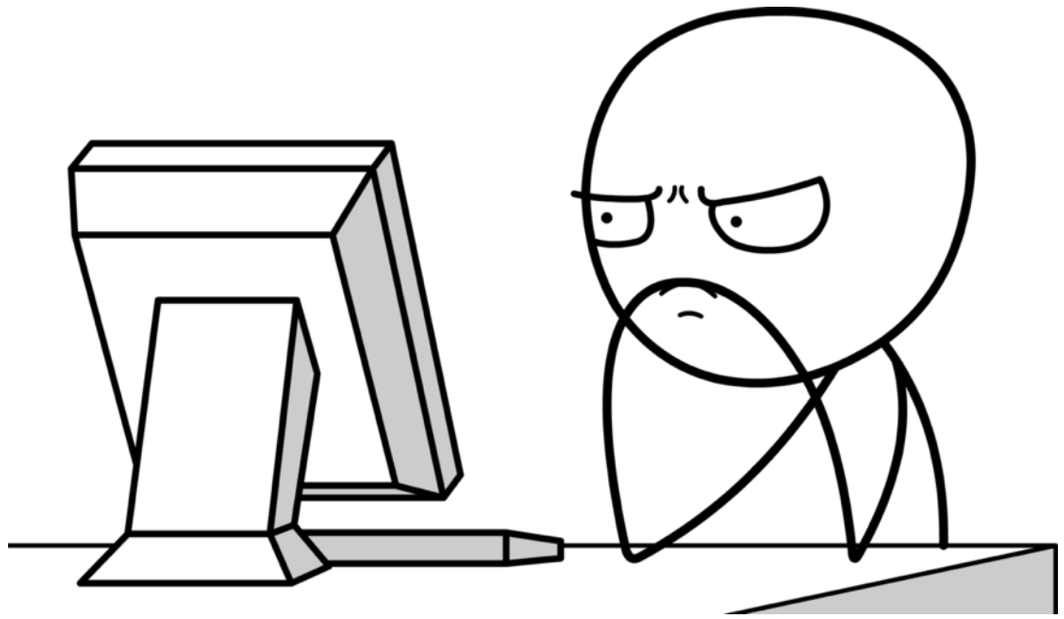
- Some documents are out of date.
- Few code examples.
- Time consuming.



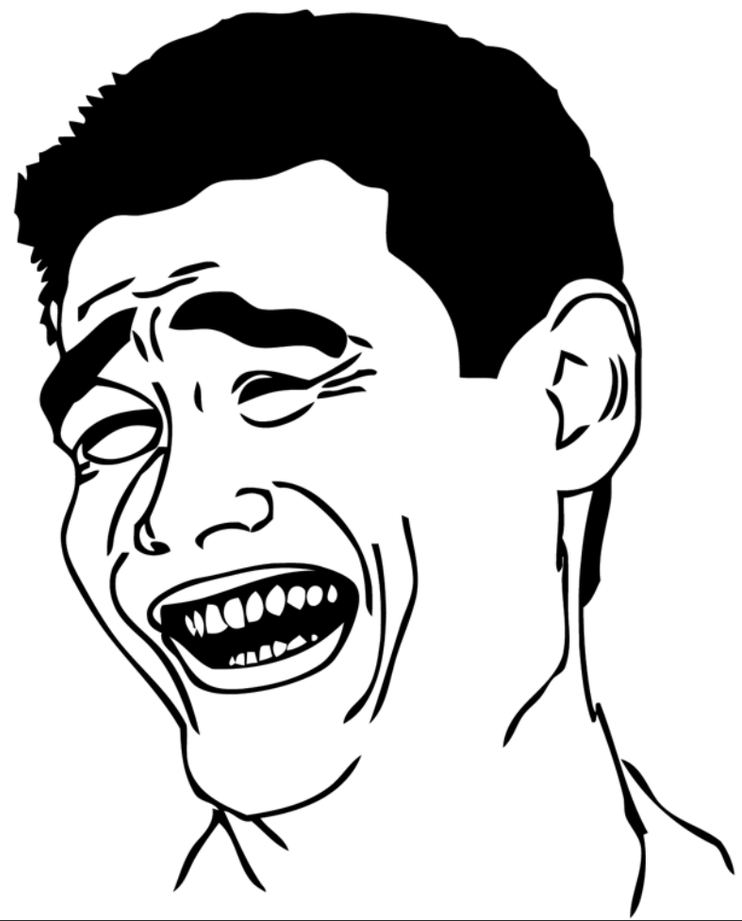
- High quality source code.
- Many, many…



- 87% of all Android API classes are post
- 56% covered by code examples
- 77% of all Java API classes

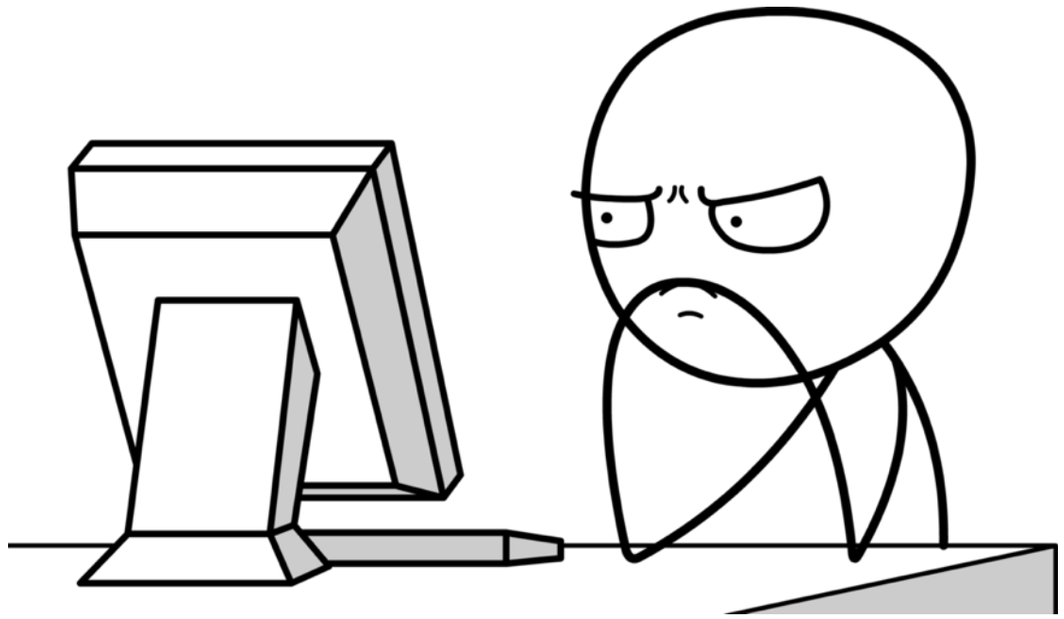


- 87% of all Android API classes are post
- 56% covered by code examples
- 77% of all Java API classes
- Good resource.



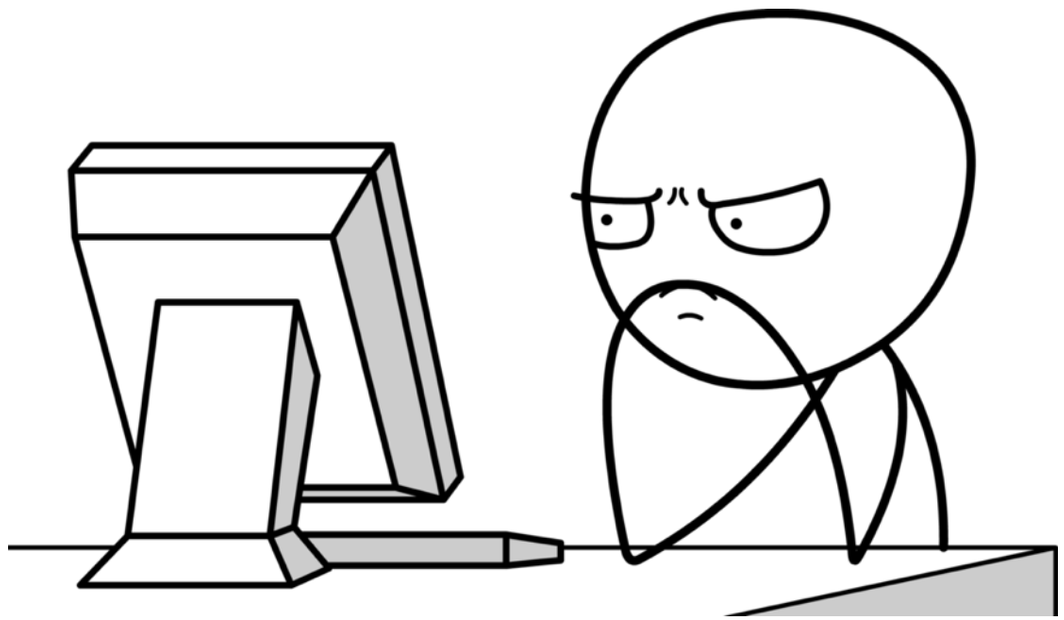
However…

- Most answers have code snippets (65%).
- Most snippets are not complete files (83%).
- Missing class or method declaration.
- Treats snippets as plain text (lexical search only).



Problems…

- Methods in different classes share similar names.
- Types are implicitly used but are never explicitly named.
- Type usage examples can be lost



For example…

```
public View getView(final int position,
    View convertView, final ViewGroup parent) {
        ...
        parent.getChildAt(position)
                .findViewById(R.id.progressbar_Horizontal)
                .setVisibility(View.VISIBLE);
        ...
}
```

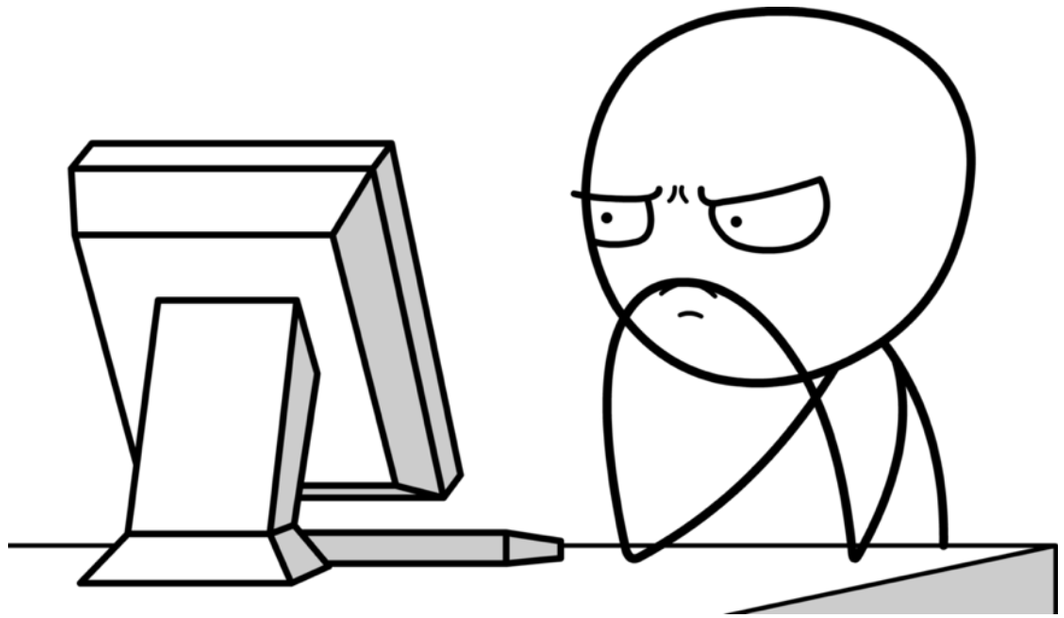
- 6 methods named setVisibility:

```
android.view.View.setVisibility(int)
android.widget.ProgressBar.setVisibility(int)
android.app.MediaRouteButton.setVisibility(int)
android.support.v7.app.MediaRouteButton.setVisibility(int)
android.view.SurfaceView.setVisibility(int)
android.widget.ImageView.setVisibility(int)
```

- Type information goes unnoticed.

An Extreme Example…

Method named describeContents in Android API…

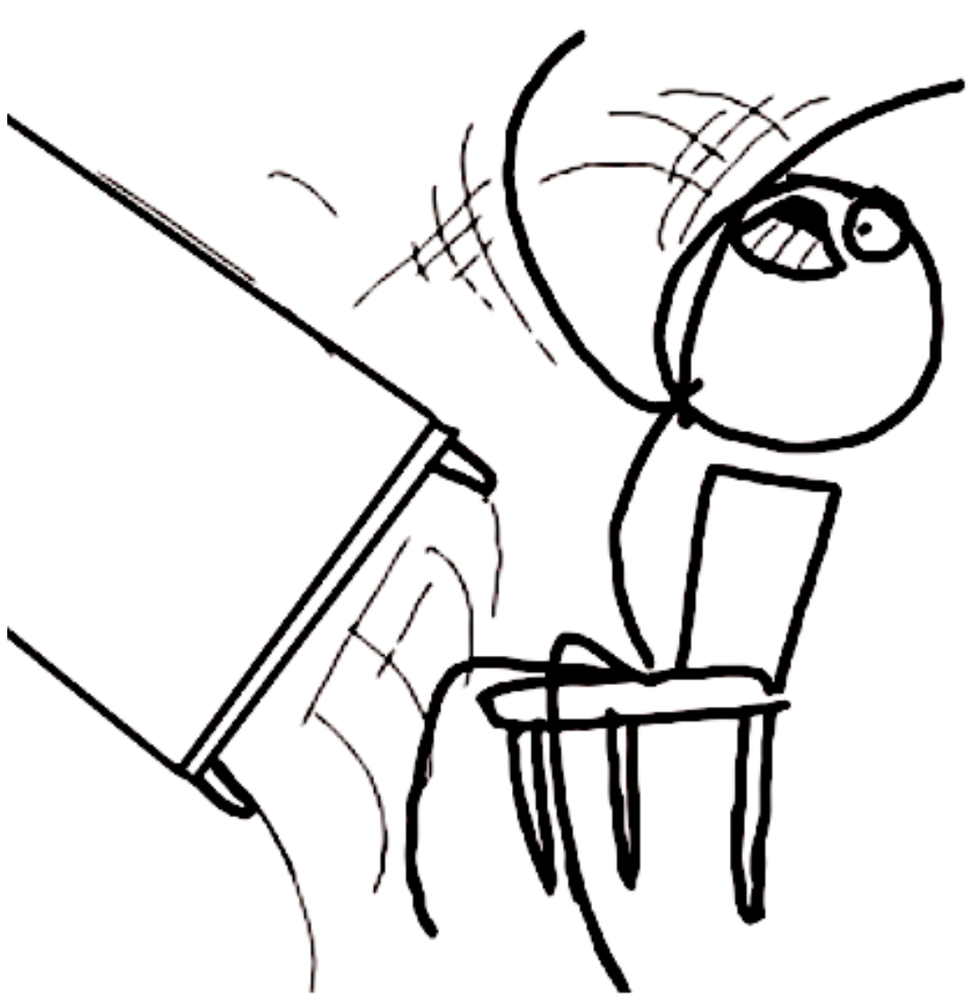


An Extreme Example…

Method named describeContents in Android API…

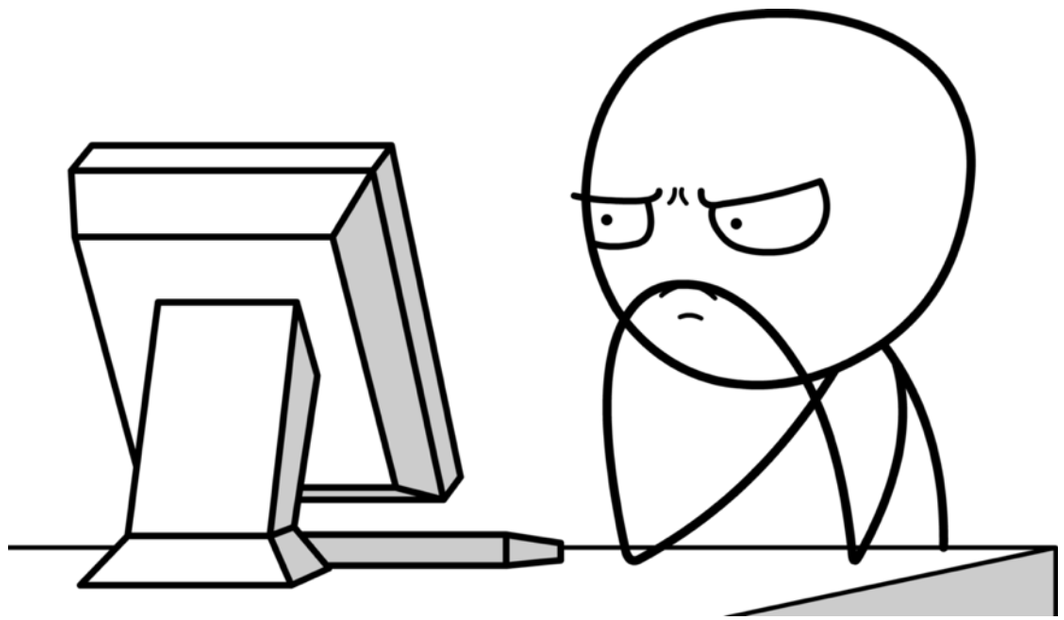
Android – android.content.IntentFilter
Android – android.content.ContentValues
Android – android.content.ContentProviderResult
Android – android.content.[illegible]ProviderOperation
Android – android.content.[illegible]
Android – android.content.[illegible]IconResource
Android – android.content.pm.PackageInfo
Android – android.content.pm.ResolveInfo
Android – android.content.SyncStats
Android – android.content.SyncAdapterType
Android – android.content.pm.Signature
Android – android.content.pm.ActivityInfo
Android – android.content.pm.ServiceInfo
Android – android.content.pm.PackageStats
Android – android.content.PeriodicSync
Android – android.content.UriPermission
Android – android.content.RestrictionEntry
Android – android.content.pm.ConfigurationInfo
Android – android.content.pm.PermissionGroupInfo
Android – android.content.pm.ApplicationInfo
Android – android.content.SyncResult
Android – android.content.pm.InstrumentationInfo
Android – android.content.pm.PermissionInfo
Android – android.content.res.ObbInfo
Android – android.content.res.Configuration
Android – android.content.pm.ProviderInfo
Android – android.content.pm.FeatureInfo
Android – android.content.SyncRequest
Android – android.content.res.AssetFileDescriptor
Android – android.content.res.ColorStateList
Android – android.gesture.Gesture

200 types!



Lexical search…

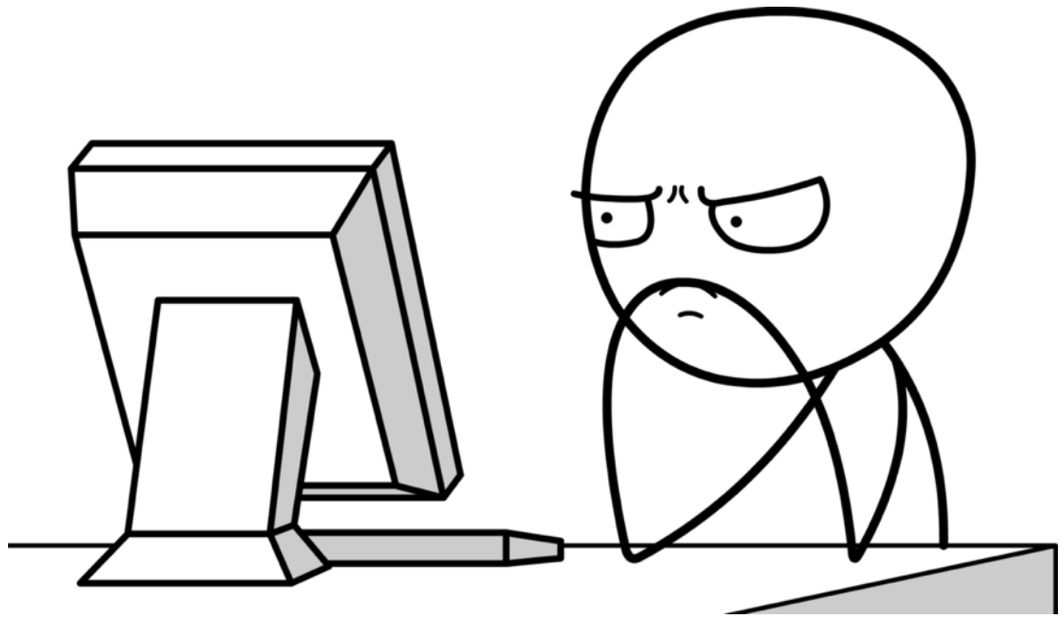
- Fail to utilize any structural information.
- Mask good API usage examples.
- Difficult for a user to identify relevant results



Ideally we prefer to…

- More than just lexical search
- Identify all the relevant information about the API usage.
- Use structural information

```
android.view.ViewGroup.getChildAt(int)
                       ↓
android.view.View.findViewById(int)
                       ↓
android.view.View.setVisibility(int)
```



Approach

Properties of snippets…

- Incomplete
- Extend on details provided in the question
- Often skip certain aspects (like variable declarations)
- Often with explanations
- Spread across several code blocks
- Complicated to analysis

In your layout:

```
<com.example.stackoverflow.MapImageView
    android:id="@+id/img_map"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/map"
    android:scaleType="fitXY" />
```

Note that `scaleType="fitXY"` is important, because coordinates are mapped to ImageView bounds, not the map picture.

In your activity:

```
MapImageView mapView = (MapImageView)findViewById(R.id.img_map);
mapView.setBounds(52.734778f, 19.979026f, -130.78125f, -62.402344f);

mapView.addPoint(42.163403f,-70.839844f);
mapView.addPoint(42.163403f,-70.839844f);
```

Result:

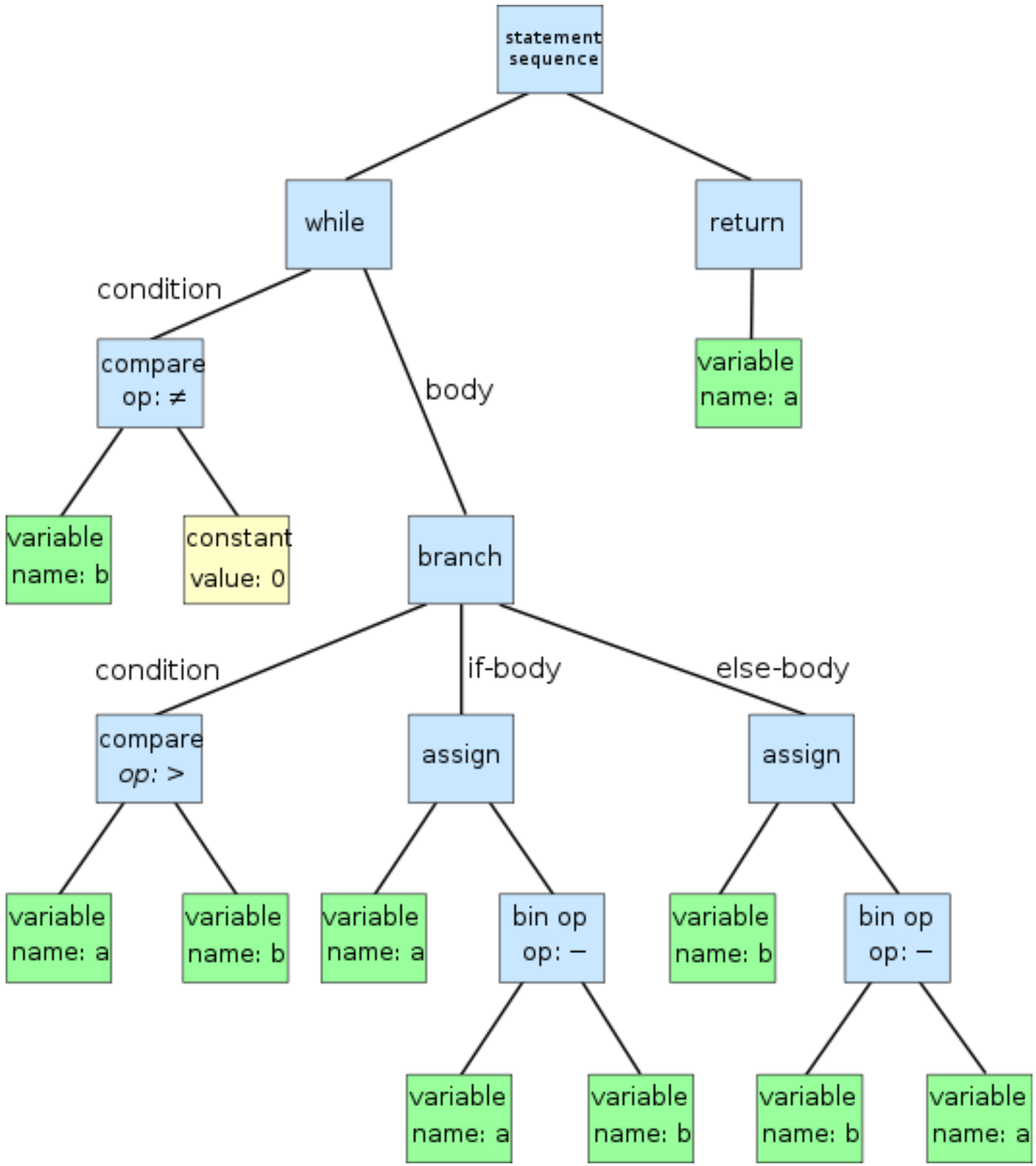
Their approach…

- A partial program analysis framework.
  - Works on arbitrarily small code fragments
- A simple oracle that describes the API space
- Two Steps:
  - Parsing Snippets
  - Inferring API Usage
- Case study on Android.

Parsing Snippets

- Constructs an Abstract Syntax Tree (AST) for each snippet:
  - Each code block as a snippet
  - Greater than 2 LOC
  - Marked as solutions

- Use Eclipse JDT with DOM representation of AST
  - Wrap free standing snippets before parsing


statement sequence
while
return
condition
compare op: ≠
body
variable name: a
variable name: b
constant value: 0
branch
condition
if-body
else-body
compare op: >
assign
assign
variable name: a
variable name: b
variable name: a
bin op op: −
variable name: b
bin op op: −
variable name: a
variable name: b
variable name: b
variable name: a

Inferring API Usage

A simple set of heuristic

- Traverse the AST for type information
- Use type information to resolve method calls
  - Oracle gives list of candidates
  - Annotate invocation in the AST with candidates
- A list of candidate return types for chained method invocations
- Similar techniques for anonymous class declaration
- Identify overridden methods to infer interfaces and superclasses

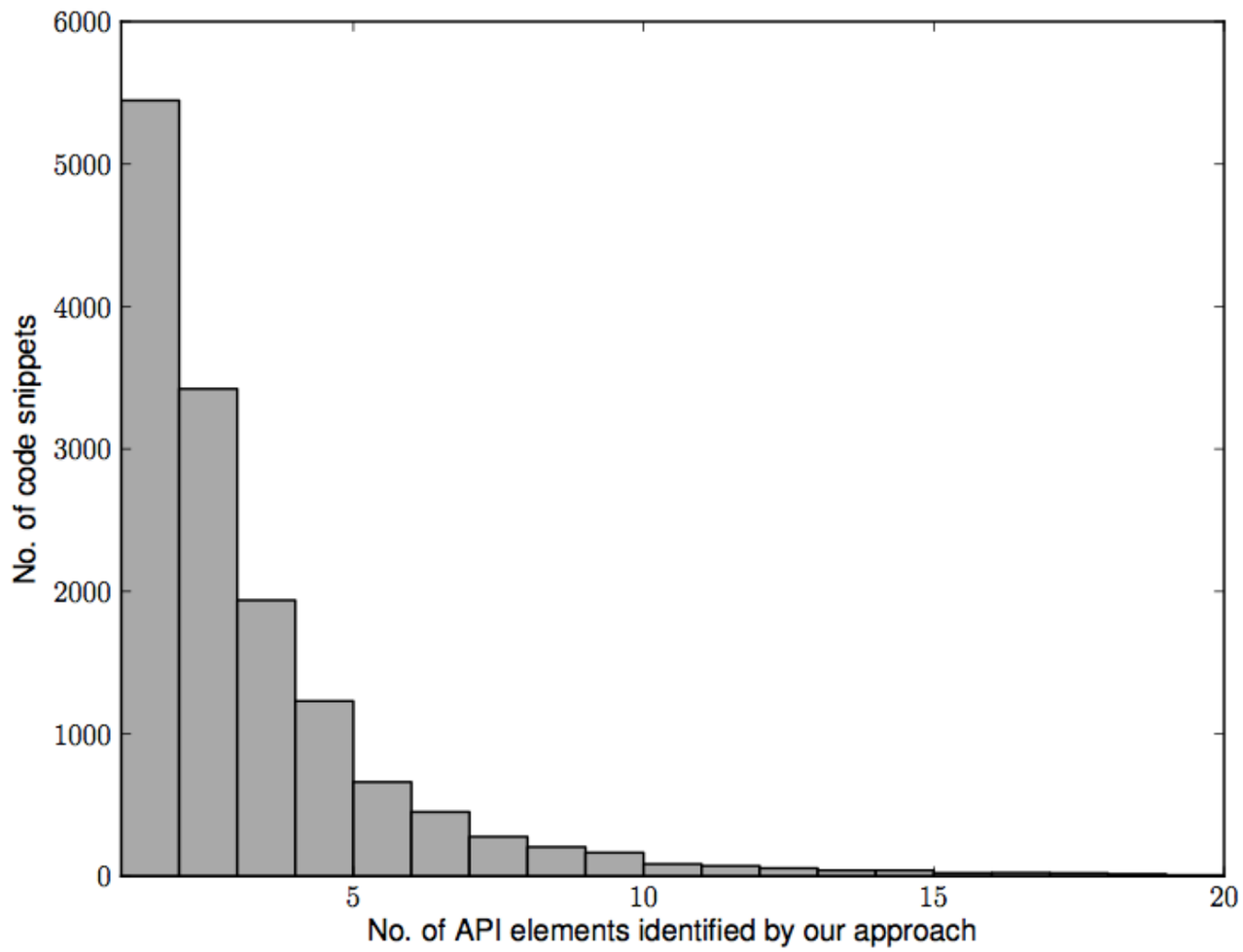
- If a method has no corresponding reference
  - Predict
  - Provide a list of all possible candidates

Evaluate

Experiment

- Android-tagged posts
- 21,250 source code snippets
- 253,137 API classes and methods
  - 75,388 API methods
  - 17,7799 API classes

- 75,338/75,338 methods with only one candidate
- 17,799/17,799 classes with only one candidate



Yields an 100% API match

Snippet Search Data

Analyze the most commonly used API elements in Android

TABLE I: Most-referenced Android API Types

API Type	Count
android.content.Intent	10550
android.view.View	8519
android.widget.TextView	5621
android.app.Activity	5473
android.os.Bundle	4503

TABLE II: Most-referenced Android API Methods

API Method	Count
android.view.View.findViewById(int)	1257
android.app.Activity.onCreate(android.os.Bundle)	1177
android.app.Activity.findViewById(int)	1174
android.util.Log.d(java.lang.String, java.lang.String)	1161
android.widget.Toast.show()	1063

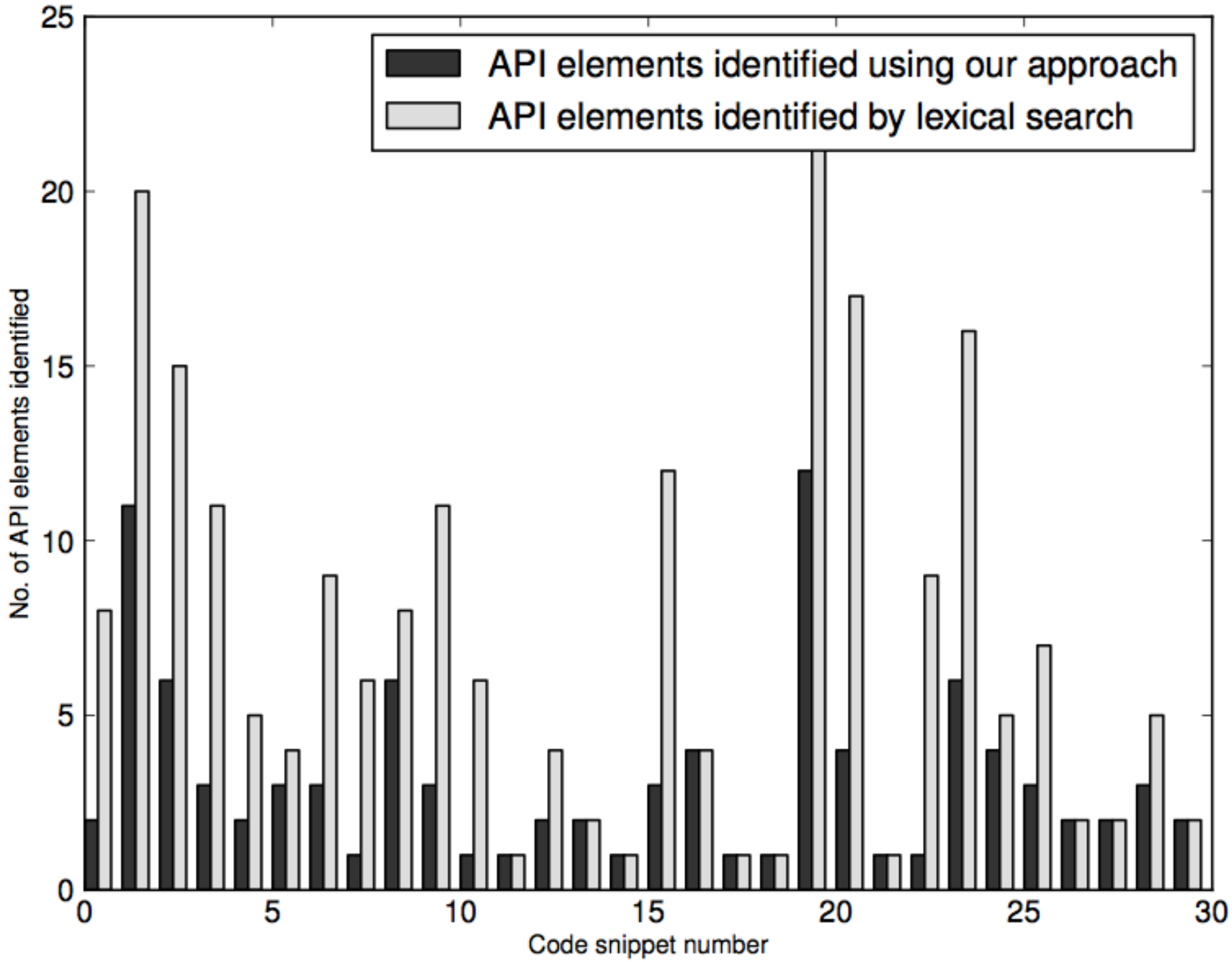
Snippet Search Data

- 24,545 total method declarations (excluding constructors)
  - Only 6,720 are unique
  - 1,7825 clashes with on average 33 others

- 23,239 instances could not fully disambiguate

Comparison

- Randomly select 30 code snippets
- Identify the exact method names
- Compare with lexical approach



Decrease mis-reported results at most 51%

Conclusion

- Accepted solution are more like to find “best practice” usage
- Their approach effectively identify API usage