10 TECHNIQUES TO UNDERSTAND CODE YOU DON'T KNOW

Jonathan Boccara

@JoBoccara

Fluent {C++}

Good developers can...

Write good code and Read existing bad code

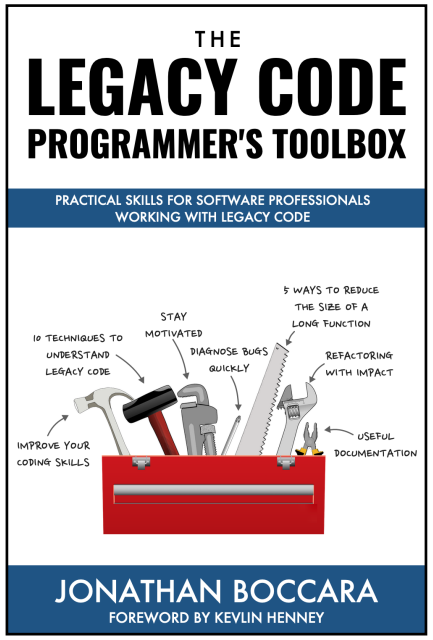
that makes you feel like running away but you can't

because it also makes you going to be sick

Hi, I'm Jonathan Boccara!

@JoBoccara

ABOUT WORKING WITH EXISTING CODE



Fluent {C++}

EXPRESSIVE CODE IN C++

ABOUT WRITING GOOD CODE

I WORK THERE

MUREX™

FINANCIAL SOFTWARE

10 TECHNIQUES TO UNDERSTAND CODE YOU DON'T KNOW

OUTLINE:

- Exploring the code (3 techniques)
- Becoming a code speed-reader (4 techniques)
- Understanding code in details (3 techniques)

Exploring the code

(3 techniques)

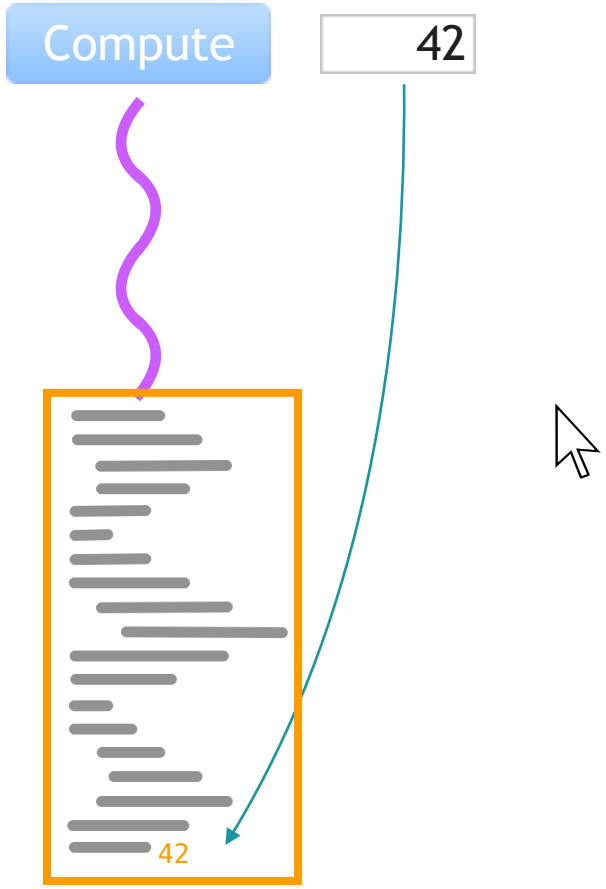
TECHNIQUE #1

Know your I/O frameworks.


Compute
42


Compute

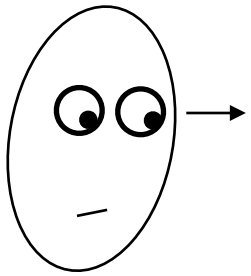

Compute


Compute
42
42

UI Framework

Get familiar with the code of your UI framework

UI Framework
REST handler
Logs
Unit tests



I/O Framework

TECHNIQUE #2

Find a stronghold.

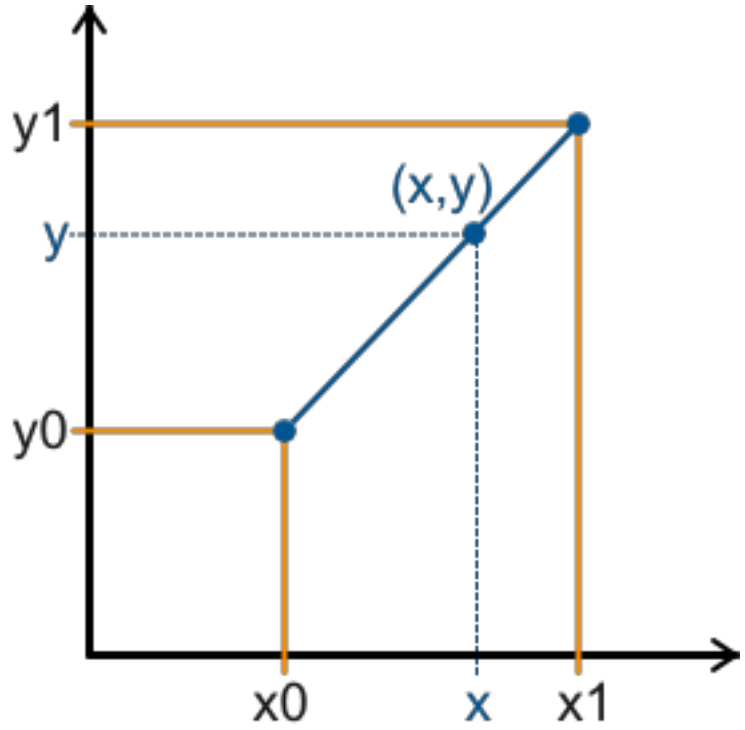
(Legend: dark = code you don’t know)

WTF

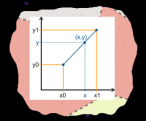
map = codebase

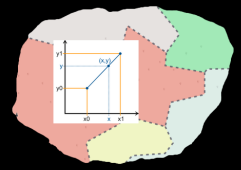
stronghold = code you understand perfectly well
even one line

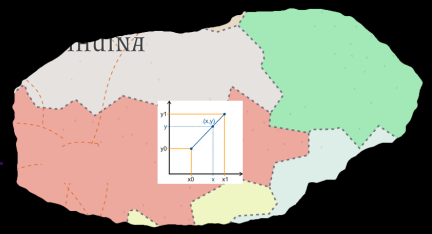
Chances are you can figure out the immediate surroundings too.



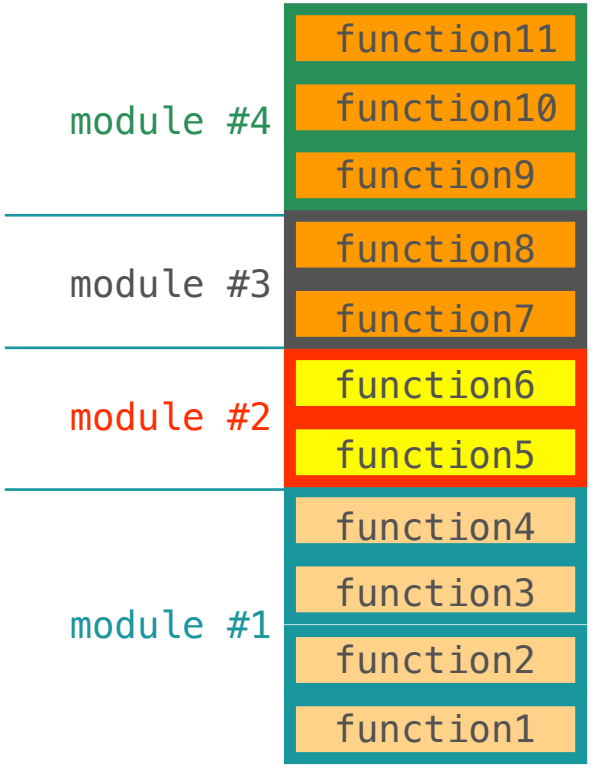
$$y = y0 + (x - x0) \times \frac{y1 - y0}{x1 - x0}$$


y1
y
y0
(x,y)
x0
x
x1


y1
y
y0
(x,y)
x0
x
x1

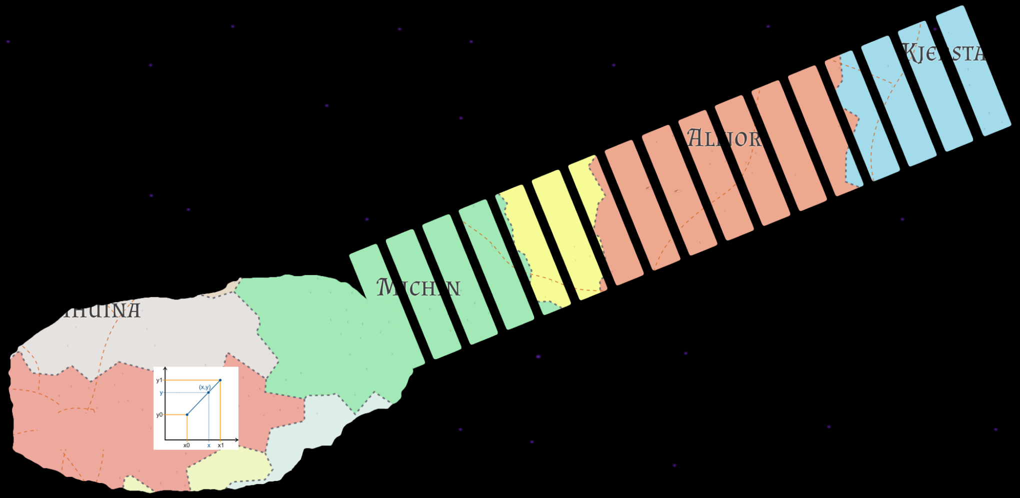

y1
y
y0
(x,y)
x0
x
x1

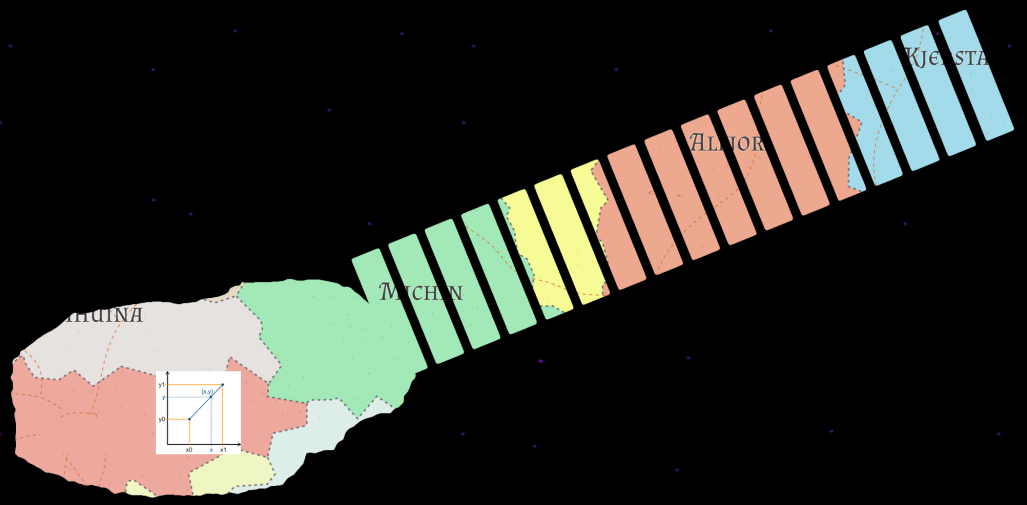
TECHNIQUE #3



Analyse call stacks.

y1
y
y0
x0
x
x1


y1
y
y0
(x,y)
x0
x
x1


Kjé sta
Allior
Michon

What is an interesting stack?

- A deep stack
- A common use case of the application

What is an interesting stack?

- A deep stack
- A common use case of the application

How to find an interesting stack?

- Your dev lead
- Business people for the common use case
- Your knowledge of the I/O to locate it

Flamegraphs

Exploring the code (3 techniques)

TECHNIQUE #1: Know your I/O frameworks.

TECHNIQUE #2: Find a stronghold.

TECHNIQUE #3: Analyse call stacks.

10 TECHNIQUES TO UNDERSTAND CODE YOU DON'T KNOW

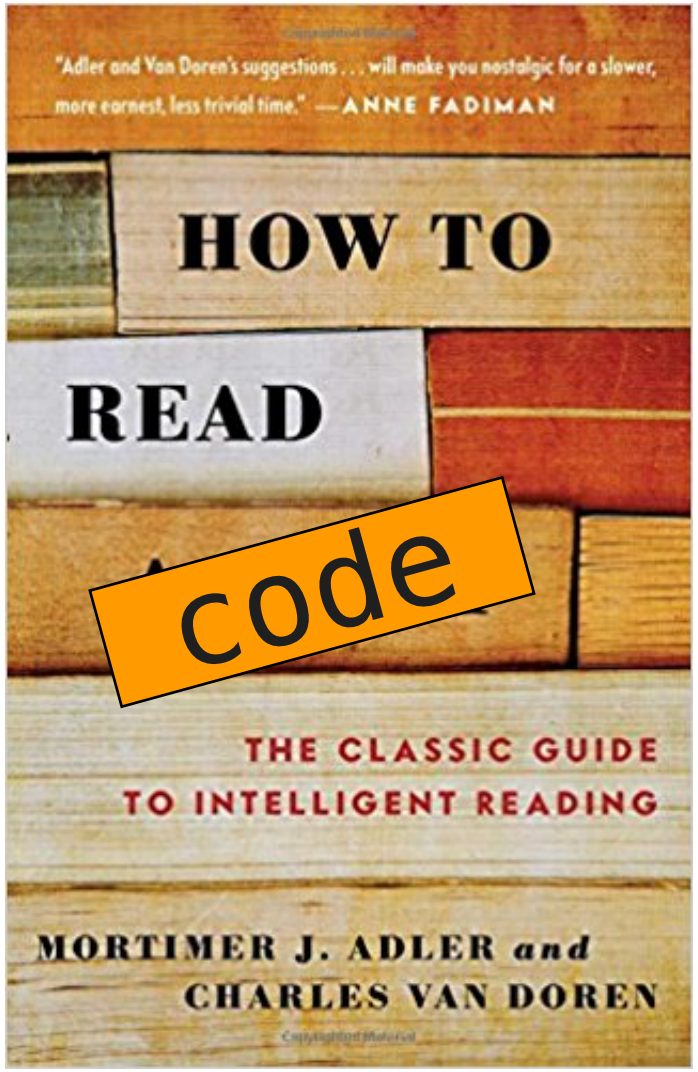
OUTLINE:

- Exploring the code (3 techniques)
- Becoming a code speed-reader (4 techniques)
- Understanding code in details (3 techniques)

Becoming a code speed-reader

(4 techniques)


doSomething()
A 2000-line function that will
make your mind bend



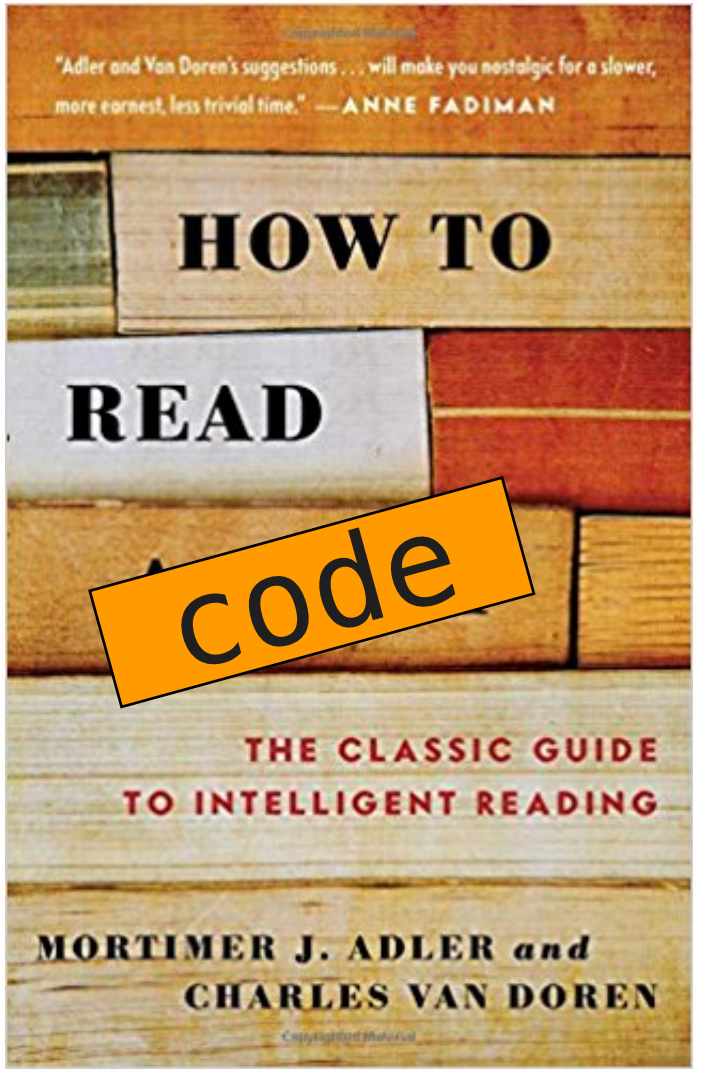
Don't read
cover to cover

Goal is not to
understand *everything*

Focus on where the
information is

TECHNIQUE #4

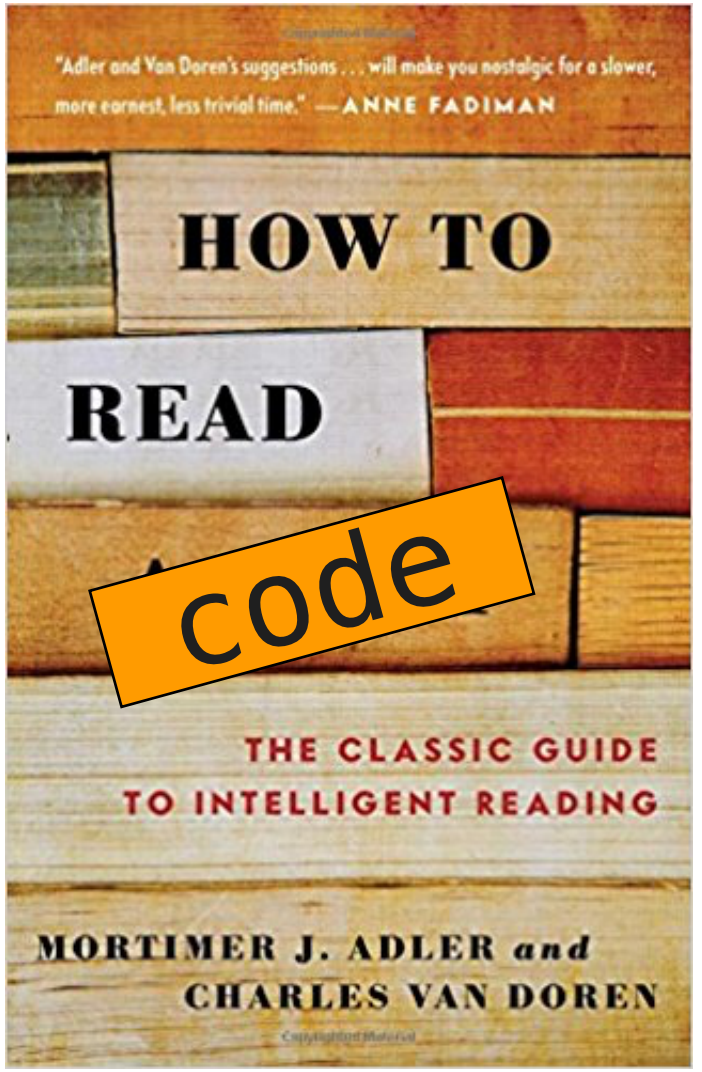
Start reading
from the end.



TECHNIQUE #4

Information is
at the beginning
and the end

beginning = prototype
end = ?



```
public List<MultipleAlignmentBlock> loadAlignments(String chr, int start, int end) throws IOException {

    IntervalTree ivTree = index.getIntervalTree(chr);
    if (ivTree == null) return null;

    List<Interval> intervals = ivTree.findOverlapping(start, end);
    if (intervals.isEmpty()) {
        return null;
    }

    // Find the starting (left most) interval.  Alignment blocks do not overlap, so we can start at the
    // minimum file offset and just proceed until the end of the interval.
    long startPosition = Long.MAX_VALUE;
    for (Interval iv : intervals) {
        startPosition = Math.min(startPosition, iv.getValue());
    }

    SeekableStream is = null;

    is = IGVSeekableStreamFactory.getInstance().getStreamFor(path);
    is.seek(startPosition);

    BufferedReader reader = new BufferedReader(new InputStreamReader(is), 256000);

    List<MultipleAlignmentBlock> alignments = new ArrayList<MultipleAlignmentBlock>();

    String line;
    while ((line = reader.readLine()) != null) {
        if (line.startsWith("a ")) {
            // TODO -- parse score (optional)
            MultipleAlignmentBlock block = parseBlock(reader);
            if(block.getEnd() < start) {
                continue;
            }
            if (block.getStart() > end || !block.getChr().equals(chr)) {
                break;
            } else {
                alignments.add(block);
            }
        }
    }
    return alignments;
}
```

TECHNIQUE #4

Start reading
from the end.

outputs.

- Returned value
- Out parameter
- Data member
- IO
- Global variable
- OUTPUT VIA EXCEPTION

Heuristics: look for outputs towards the end

TECHNIQUE #5

Find the frequent words.

```
bool CSetting::ReadValue( CRegKey &regKey, const wchar_t *valName )
{
    // bool, int, hotkey, color
    if (type==CSetting::TYPE_BOOL || (type==CSetting::TYPE_INT && this[1].type!=CSetting::TYPE_RADIO) || type==CSetting::TYPE_HOTKEY || type==CSetting::TYPE_HOTKEY_ANY || type==CSetting::TYPE_COLOR)
    {
        DWORD val;
        if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
        {
            if (type==CSetting::TYPE_BOOL)
                value=CComVariant(val?1:0);
            else
                value=CComVariant((int)val);
            return true;
        }
        return false;
    }
   // radio
    if (type==CSetting::TYPE_INT && this[1].type==CSetting::TYPE_RADIO)
    {
        ULONG len;
        DWORD val;
        if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            CString text;
            regKey.QueryStringValue(valName,text.GetBuffer(len),&len);
            text.ReleaseBuffer(len);
            val=0;
            for (const CSetting *pRadio=this+1;pRadio->type==CSetting::TYPE_RADIO;pRadio++,val++)
            {
                if (_wcsicmp(text,pRadio->name)==0)
                {
                    value=CComVariant((int)val);
                    return true;
                }
            }
        }
        else if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
        {
            value=CComVariant((int)val);
            return true;
        }
        return false;
    }
   // string
    if (type>=CSetting::TYPE_STRING && type<CSetting::TYPE_MULTISTRING)
    {
        ULONG len;
        if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len-1);
            regKey.QueryStringValue(valName,value.bstrVal,&len);
            return true;
        }
        return false;
    }
   // multistring
    if (type==CSetting::TYPE_MULTISTRING)
    {
        ULONG len;
        if (regKey.QueryMultiStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len-1);
            regKey.QueryMultiStringValue(valName,value.bstrVal,&len);
            for (int i=0;i<(int)len-1;i++)
                if (value.bstrVal[i]==0)
                    value.bstrVal[i]='\n';
            return true;
        }
        else if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len);
            regKey.QueryStringValue(valName,value.bstrVal,&len);
            if (len>0)
            {
                value.bstrVal[len-1]='\n';
                value.bstrVal[len]=0;
            }
            return true;
        }
        return false;
    }
    Assert(0);
    return false;
}
```

Word	# occurrences
len	20
value	17
CSetting	15
if	14
type	13
regKey	11
valName	11
return	11
val	10
bstrVal	10
1	8
0	8
NULL	7
true	6
QueryStringValue	6
ERROR_SUCCESS	6
int	6
…	

```
bool CSetting::ReadValue( CRegKey &regKey, const wchar_t *valName )
{
    // bool, int, hotkey, color
    if (type==CSetting::TYPE_BOOL || (type==CSetting::TYPE_INT && this[1].type!=CSetting::TYPE_RADIO) || type==CSetting::TYPE_HOTKEY || type==CSetting::TYPE_HOTKEY_ANY || type==CSetting::TYPE_COLOR)
    {
        DWORD val;
        if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
        {
            if (type==CSetting::TYPE_BOOL)
                value=CComVariant(val?1:0);
            else
                value=CComVariant((int)val);
            return true;
        }
        return false;
    }
   // radio
    if (type==CSetting::TYPE_INT && this[1].type==CSetting::TYPE_RADIO)
    {
        ULONG len;
        DWORD val;
        if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            CString text;
            regKey.QueryStringValue(valName,text.GetBuffer(len),&len);
            text.ReleaseBuffer(len);
            val=0;
            for (const CSetting *pRadio=this+1;pRadio->type==CSetting::TYPE_RADIO;pRadio++,val++)
            {
                if (_wcsicmp(text,pRadio->name)==0)
                {
                    value=CComVariant((int)val);
                    return true;
                }
            }
        }
        else if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
        {
            value=CComVariant((int)val);
            return true;
        }
        return false;
    }
   // string
    if (type>=CSetting::TYPE_STRING && type<CSetting::TYPE_MULTISTRING)
    {
        ULONG len;
        if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len-1);
            regKey.QueryStringValue(valName,value.bstrVal,&len);
            return true;
        }
        return false;
    }
   // multistring
    if (type==CSetting::TYPE_MULTISTRING)
    {
        ULONG len;
        if (regKey.QueryMultiStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len-1);
            regKey.QueryMultiStringValue(valName,value.bstrVal,&len);
            for (int i=0;i<(int)len-1;i++)
                if (value.bstrVal[i]==0)
                    value.bstrVal[i]='\n';
            return true;
        }
        else if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len);
            regKey.QueryStringValue(valName,value.bstrVal,&len);
            if (len>0)
            {
                value.bstrVal[len-1]='\n';
                value.bstrVal[len]=0;
            }
            return true;
        }
        return false;
    }
    Assert(0);
    return false;
}
```

inputs

Word	# occurrences
len	20
value	17
CSetting	15
if	14
type	13
regKey	11
valName	11
return	11
val	10
bstrVal	10
1	8
0	8
NULL	7
true	6
QueryStringValue	6
ERROR_SUCCESS	6
int	6
…	

```
IatHookData *SetIatHook( IMAGE_DOS_HEADER *dosHeader, DWORD iatOffset, DWORD intOfset, const char *targetProc, void *newProc)
{
    IMAGE_THUNK_DATA *thunk=(IMAGE_THUNK_DATA*)PtrFromRva(dosHeader,iatOffset);
    IMAGE_THUNK_DATA *origThunk=(IMAGE_THUNK_DATA*)PtrFromRva(dosHeader,intOfset);
    for (;origThunk->u1.Function;origThunk++,thunk++)
    {
        if (origThunk->u1.Ordinal&IMAGE_ORDINAL_FLAG)
        {
            if (IS_INTRESOURCE(targetProc) && IMAGE_ORDINAL(origThunk->u1.Ordinal)==(uintptr_t)targetProc)
                break;
        }
        else
        {
            IMAGE_IMPORT_BY_NAME *import=(IMAGE_IMPORT_BY_NAME*)PtrFromRva(dosHeader,origThunk->u1.AddressOfData);
            if (!IS_INTRESOURCE(targetProc) && strcmp(targetProc,(char*)import->Name)==0)
                break;
        }
    }
    if (origThunk->u1.Function)
    {
        IatHookData *hook=g_IatHooks+g_IatHookCount;
        g_IatHookCount++;
        hook->jump[0]=hook->jump[1]=0x90; // NOP
        hook->jump[2]=0xFF; hook->jump[3]=0x25; // JUMP
#ifdef _WIN64
        hook->jumpOffs=0;
#else
        hook->jumpOffs=(DWORD)(hook)+8;
#endif
        hook->newProc=newProc;
        hook->oldProc=(void*)thunk->u1.Function;
        hook->thunk=thunk;
        DWORD oldProtect;
        VirtualProtect(&thunk->u1.Function,sizeof(void*),PAGE_READWRITE,&oldProtect);
        thunk->u1.Function=(DWORD_PTR)hook;
        VirtualProtect(&thunk->u1.Function,sizeof(void*),oldProtect,&oldProtect);
        return hook;
    }
    return NULL;
}
```

```
int CSettingsParser::ParseTreeRec( const wchar_t *str, std::vector<TreeItem> &items, CString *names, int level )
{
    size_t start=items.size();
    while (*str)
    {
        wchar_t token[256];
        str=GetToken(str,token,_countof(token),L", \t");
        if (token[0])
        {
            //
            bool bFound=false;
            for (int i=0;i<level;i++)
                if (_wcsicmp(token,names[i])==0)
                {
                    bFound=true;
                    break;
                }
                if (!bFound)
                {
                    TreeItem item={token,-1};
                    items.push_back(item);
                }
        }
    }
    size_t end=items.size();
    if (start==end) return -1;

    TreeItem item={L"",-1};
    items.push_back(item);

    if (level<MAX_TREE_LEVEL-1)
    {
        for (size_t i=start;i<end;i++)
        {
            wchar_t buf[266];
            Sprintf(buf,_countof(buf),L"%s.Items",items[i].name);
            const wchar_t *str2=FindSetting(buf);
            if (str2)
            {
                names[level]=items[i].name;
                // these two statements must be on separate lines. otherwise items[i] is evaluated before ParseTreeRec, but
                // the items vector can be reallocated inside ParseTreeRec, causing the address to be invalidated -> crash!
                int idx=ParseTreeRec(str2,items,names,level+1);
                items[i].children=idx;
            }
        }
    }
    return (int)start;
}
```

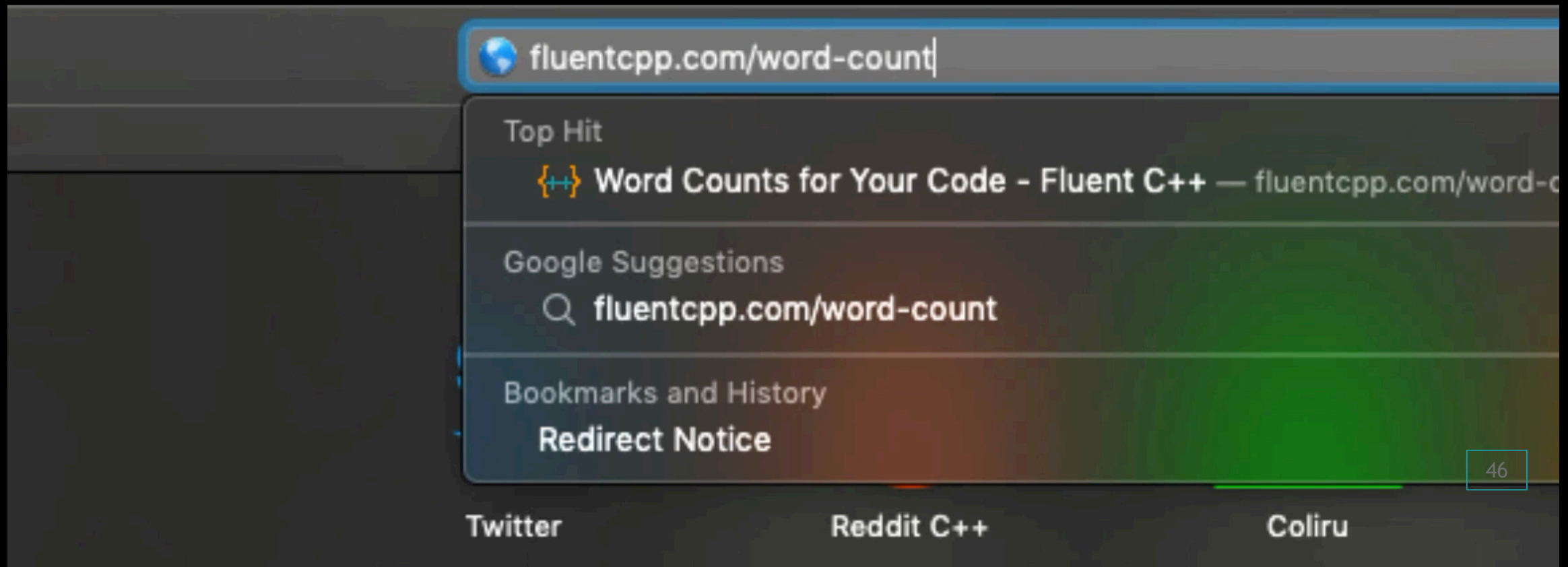
Word	#	span	proportion
items	11	44	85 %
i	11	33	63 %
token	6	15	29 %
if	6	31	60 %
1	5	24	46 %
level	5	43	83 %
int	5	48	92 %
names	4	43	83 %
str	4	7	13 %
wchar_t	4	37	71 %
ParseTreeRec	4	43	83 %
start	4	46	88 %
item	4	10	19 %
buf	4	3	6 %

Tools:

- Your eyes
- IDE search highlighting
- Online tool

Tools:

- Your eyes
- IDE search highlighting
- Online tool



VARIOUS WAYS TO COUNT WORDS

- entire words
- wordsInCamelCase
- Case insensitive

COUNTING THE WORDS OF A MODULE

- Whole files
- Questions for newcomers

TECHNIQUE #6

Filter on control flow.

```
else
if          switch
  for  while
          case
do
      try
catch
```

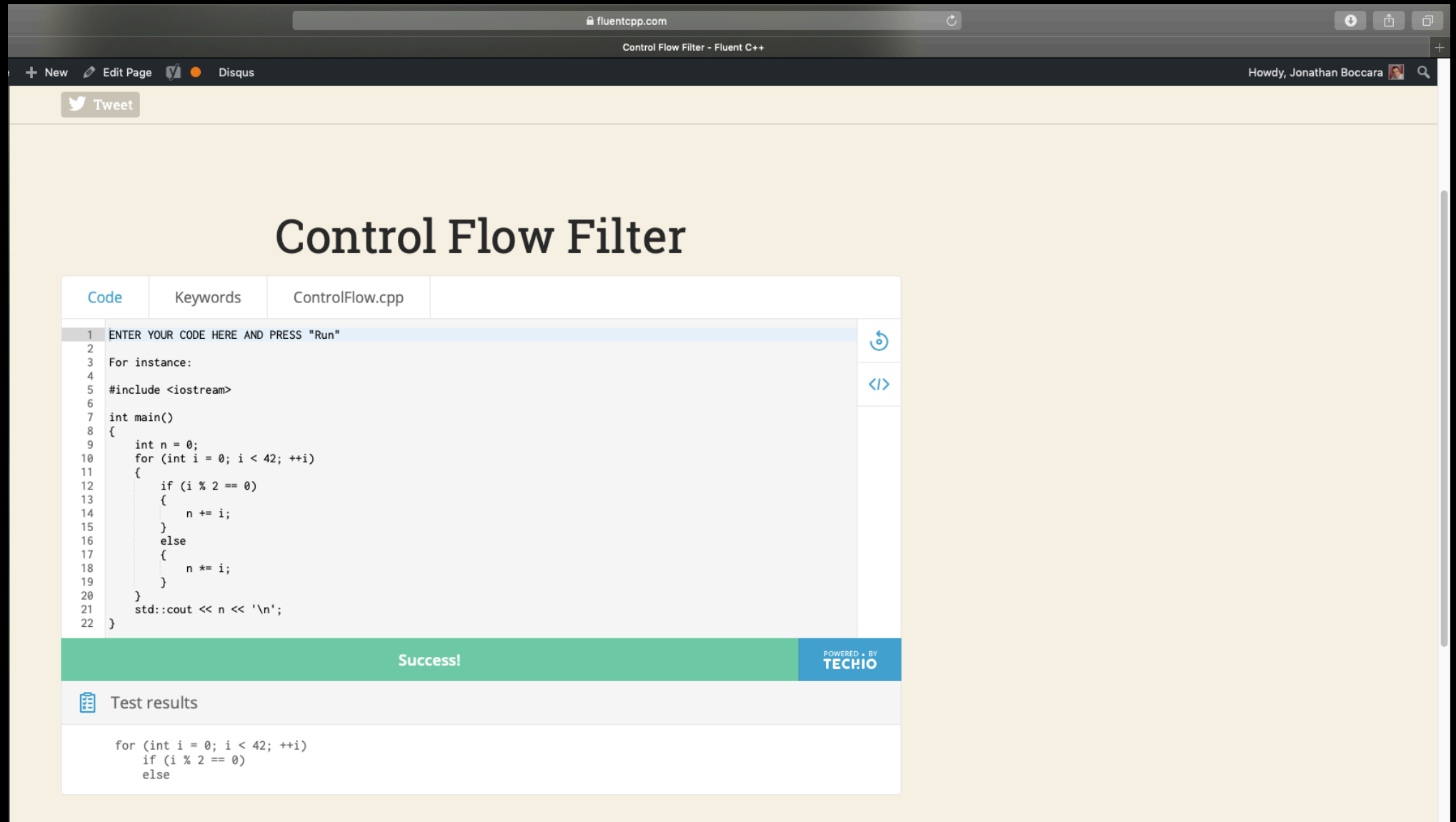
```
bool CSetting::ReadValue( CRegKey &regKey, const wchar_t *valName )
{
    // bool, int, hotkey, color
    if (type==CSetting::TYPE_BOOL || (type==CSetting::TYPE_INT && this[1].type!=CSetting::TYPE_RADIO) || type==CSetting::TYPE_HOTKEY || type==CSetting::TYPE_HOTKEY_ANY || type==CSetting::TYPE_COLOR)
    {
        DWORD val;
        if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
        {
            if (type==CSetting::TYPE_BOOL)
                value=CComVariant(val?1:0);
            else
                value=CComVariant((int)val);
            return true;
        }
        return false;
    }
    // radio
    if (type==CSetting::TYPE_INT && this[1].type==CSetting::TYPE_RADIO)
    {
        ULONG len;
        DWORD val;
        if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            CString text;
            regKey.QueryStringValue(valName,text.GetBuffer(len),&len);
            text.ReleaseBuffer(len);
            val=0;
            for (const CSetting *pRadio=this+1;pRadio->type==CSetting::TYPE_RADIO;pRadio++,val++)
            {
                if (_wcsicmp(text,pRadio->name)==0)
                {
                    value=CComVariant((int)val);
                    return true;
                }
            }
        }
        else if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
        {
            value=CComVariant((int)val);
            return true;
        }
        return false;
    }
    // string
    if (type>=CSetting::TYPE_STRING && type<CSetting::TYPE_MULTISTRING)
    {
        ULONG len;
        if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len-1);
            regKey.QueryStringValue(valName,value.bstrVal,&len);
            return true;
        }
        return false;
    }
    // multistring
    if (type==CSetting::TYPE_MULTISTRING)
    {
        ULONG len;
        if (regKey.QueryMultiStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len-1);
            regKey.QueryMultiStringValue(valName,value.bstrVal,&len);
            for (int i=0;i<(int)len-1;i++)
                if (value.bstrVal[i]==0)
                    value.bstrVal[i]='\n';
            return true;
        }
        else if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        {
            value.vt=VT_BSTR;
            value.bstrVal=SysAllocStringLen(NULL,len);
            regKey.QueryStringValue(valName,value.bstrVal,&len);
            if (len>0)
            {
                value.bstrVal[len-1]='\n';
                value.bstrVal[len]=0;
            }
            return true;
        }
        return false;
    }
    Assert(0);
    return false;
}
```

```
:g!/\(\<if\>\|\<else\>\|\<for\>\|\<while\>\|\<do\>\
|\<switch\>\|\<case\>\|\<try\>\|\<catch\>\)/d
```

```
if (type==CSetting::TYPE_BOOL || (type==CSetting::TYPE_INT &&
    this[1].type!=CSetting::TYPE_RADIO) || type==CSetting::TYPE_HOTKEY ||
    type==CSetting::TYPE_HOTKEY_ANY || type==CSetting::TYPE_COLOR)
    if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
        if (type==CSetting::TYPE_BOOL)
        else
if (type==CSetting::TYPE_INT && this[1].type==CSetting::TYPE_RADIO)
    if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        for (const CSetting *pRadio=this+1;pRadio->type==CSetting::TYPE_RADIO;pRadio++,val++)
            if (_wcsicmp(text,pRadio->name)==0)
    else if (regKey.QueryDWORDValue(valName,val)==ERROR_SUCCESS)
if (type>=CSetting::TYPE_STRING && type<CSetting::TYPE_MULTISTRING)
    if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
if (type==CSetting::TYPE_MULTISTRING)
    if (regKey.QueryMultiStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        for (int i=0;i<(int)len-1;i++)
            if (value.bstrVal[i]==0)
    else if (regKey.QueryStringValue(valName,NULL,&len)==ERROR_SUCCESS)
        if (len>0)
```

CSetting	14
if	14
type	13
len	6
ERROR_SUCCESS	6
regKey	6
valName	6
1	4

fluentcpp.com/control-flow-filter



TECHNIQUE #7

Scan for the main action.

THE

80 20

RULE

THE 90 10 RULE

QUICK scan for the main action

- Speed over accuracy
- A second pass is possible

Things that are not the main action:

- Secondary variables
- Special cases
- Complicated stuff (in all likelyhood)
- …

```
public synchronized void processAlignments(String chr, List<Alignment> alignments) {

    Genome genome = GenomeManager.getInstance().getCurrentGenome();
    chr = genome == null ? chr : genome.getCanonicalChrName(chr);

    Map<Integer, InsertionMarker> insertionMap = insertionMaps.get(chr);
    if(insertionMap == null) {
        insertionMap =  Collections.synchronizedMap(new HashMap<>());
        insertionMaps.put(chr, insertionMap);
    }
    List<Integer> positions = positionsMap.get(chr);
    if(positions == null) {
        positions = new ArrayList<>();
        positionsMap.put(chr, positions);
    }
    int minLength = 0;
    if (PreferencesManager.getPreferences().getAsBoolean(SAM_HIDE_SMALL_INDEL)) {
        minLength = PreferencesManager.getPreferences().getAsInt(SAM_SMALL_INDEL_BP_THRESHOLD);
    }
   for (Alignment a : alignments) {
        AlignmentBlock[] blocks = a.getInsertions();
        if (blocks != null) {
            for (AlignmentBlock block : blocks) {

                if (block.getBases().length < minLength) continue;

                Integer key = block.getStart();
                InsertionMarker insertionMarker = insertionMap.get(key);
                if (insertionMarker == null) {
                    insertionMarker = new InsertionMarker(block.getStart(), block.getLength());
                    insertionMap.put(key, insertionMarker);
                    positions.add(block.getStart());
                } else {
                    insertionMarker.size = Math.max(insertionMarker.size, block.getLength());
                }
            }
        }
    }
    positions.addAll(insertionMap.keySet());
    positions.sort((o1, o2) -> o1 - o2);
}
```

Word	#	span	proportion
chr	8	14	33 %
insertionMap	7	34	81 %
block	7	12	29 %
positions	7	30	71 %
insertionMarker	6	7	17 %
if	6	23	55 %
null	5	26	62 %
key	3	5	12 %
InsertionMarker	3	25	60 %
Integer	3	22	52 %
minLength	3	10	24 %
put	3	23	55 %
get	3	23	55 %
genome	3	2	5 %
blocks	3	3	7 %
new	3	23	55 %
getStart	3	6	14 %

```
public synchronized void processAlignments(String chr, List<Alignment> alignments) {

    Genome genome = GenomeManager.getInstance().getCurrentGenome();
    chr = genome == null ? chr : genome.getCanonicalChrName(chr);

    Map<Integer, InsertionMarker> insertionMap = insertionMaps.get(chr);
    if(insertionMap == null) {
        insertionMap =  Collections.synchronizedMap(new HashMap<>());
        insertionMaps.put(chr, insertionMap);
    }
    List<Integer> positions = positionsMap.get(chr);
    if(positions == null) {
        positions = new ArrayList<>();
        positionsMap.put(chr, positions);
    }
    int minLength = 0;
    if (PreferencesManager.getPreferences().getAsBoolean(SAM_HIDE_SMALL_INDEL)) {
        minLength = PreferencesManager.getPreferences().getAsInt(SAM_SMALL_INDEL_BP_THRESHOLD);
    }
   for (Alignment a : alignments) {
        AlignmentBlock[] blocks = a.getInsertions();
        if (blocks != null) {
            for (AlignmentBlock block : blocks) {

                if (block.getBases().length < minLength) continue;

                Integer key = block.getStart();
                InsertionMarker insertionMarker = insertionMap.get(key);
                if (insertionMarker == null) {
                    insertionMarker = new InsertionMarker(block.getStart(), block.getLength());
                    insertionMap.put(key, insertionMarker);
                    positions.add(block.getStart());
                } else {
                    insertionMarker.size = Math.max(insertionMarker.size, block.getLength());
                }
            }
        }
    }
    positions.addAll(insertionMap.keySet());
    positions.sort((o1, o2) -> o1 - o2);
}
```

Word	#	span	proportion
chr	8	14	33 %
insertionMap	7	34	81 %
block	7	12	29 %
positions	7	30	71 %
insertionMarker	6	7	17 %
if	6	23	55 %
null	5	26	62 %
key	3	5	12 %
InsertionMarker	3	25	60 %
Integer	3	22	52 %
minLength	3	10	24 %
put	3	23	55 %
get	3	23	55 %
genome	3	2	5 %
blocks	3	3	7 %
new	3	23	55 %
getStart	3	6	14 %

```
public synchronized void processAlignments(String chr, List<Alignment> alignments) {

    Genome genome = GenomeManager.getInstance().getCurrentGenome();
    chr = genome == null ? chr : genome.getCanonicalChrName(chr);

    Map<Integer, InsertionMarker> insertionMap = insertionMaps.get(chr);
    if(insertionMap == null) {
        insertionMap =  Collections.synchronizedMap(new HashMap<>());
        insertionMaps.put(chr, insertionMap);
    }
    List<Integer> positions = positionsMap.get(chr);
    if(positions == null) {
        positions = new ArrayList<>();
        positionsMap.put(chr, positions);
    }
    int minLength = 0;
    if (PreferencesManager.getPreferences().getAsBoolean(SAM_HIDE_SMALL_INDEL)) {
        minLength = PreferencesManager.getPreferences().getAsInt(SAM_SMALL_INDEL_BP_THRESHOLD);
    }
   for (Alignment a : alignments) {
        AlignmentBlock[] blocks = a.getInsertions();
        if (blocks != null) {
            for (AlignmentBlock block : blocks) {

                if (block.getBases().length < minLength) continue;

                Integer key = block.getStart();
                InsertionMarker insertionMarker = insertionMap.get(key);
                if (insertionMarker == null) {
                    insertionMarker = new InsertionMarker(block.getStart(), block.getLength());
                    insertionMap.put(key, insertionMarker);
                    positions.add(block.getStart());
                } else {
                    insertionMarker.size = Math.max(insertionMarker.size, block.getLength());
                }
            }
        }
    }
    positions.addAll(insertionMap.keySet());
    positions.sort((o1, o2) -> o1 - o2);
}
```

@JoBoccara

Word	#	span	proportion
chr	8	14	33 %
insertionMap	7	34	81 %
block	7	12	29 %
positions	7	30	71 %
insertionMarker	6	7	17 %
if	6	23	55 %
null	5	26	62 %
key	3	5	12 %
InsertionMarker	3	25	60 %
Integer	3	22	52 %
minLength	3	10	24 %
put	3	23	55 %
get	3	23	55 %
genome	3	2	5 %
blocks	3	3	7 %
new	3	23	55 %
getStart	3	6	14 %

```
public synchronized void processAlignments(String chr, List<Alignment> alignments) {

    Genome genome = GenomeManager.getInstance().getCurrentGenome();
    chr = genome == null ? chr : genome.getCanonicalChrName(chr);

    Map<Integer, InsertionMarker> insertionMap = insertionMaps.get(chr);
    if(insertionMap == null) {
        insertionMap =  Collections.synchronizedMap(new HashMap<>());
        insertionMaps.put(chr, insertionMap);
    }
    List<Integer> positions = positionsMap.get(chr);
    if(positions == null) {
        positions = new ArrayList<>();
        positionsMap.put(chr, positions);
    }
    int minLength = 0;
    if (PreferencesManager.getPreferences().getAsBoolean(SAM_HIDE_SMALL_INDEL)) {
        minLength = PreferencesManager.getPreferences().getAsInt(SAM_SMALL_INDEL_BP_THRESHOLD);
    }
   for (Alignment a : alignments) {
        AlignmentBlock[] blocks = a.getInsertions();
        if (blocks != null) {
            for (AlignmentBlock block : blocks) {

                if (block.getBases().length < minLength) continue;

                Integer key = block.getStart();
                InsertionMarker insertionMarker = insertionMap.get(key);
                if (insertionMarker == null) {
                    insertionMarker = new InsertionMarker(block.getStart(), block.getLength());
                    insertionMap.put(key, insertionMarker);
                    positions.add(block.getStart());
                } else {
                    insertionMarker.size = Math.max(insertionMarker.size, block.getLength());
                }
            }
        }
    }
    positions.addAll(insertionMap.keySet());
    positions.sort((o1, o2) -> o1 - o2);
}
```

@JoBoccara

```
public synchronized void processAlignments(String chr, List<Alignment> alignments) {

    Genome genome = GenomeManager.getInstance().getCurrentGenome();
    chr = genome == null ? chr : genome.getCanonicalChrName(chr);

    Map<Integer, InsertionMarker> insertionMap = insertionMaps.get(chr);
    if(insertionMap == null) {
        insertionMap =  Collections.synchronizedMap(new HashMap<>());
        insertionMaps.put(chr, insertionMap);
    }
    List<Integer> positions = positionsMap.get(chr);
    if(positions == null) {
        positions = new ArrayList<>();
        positionsMap.put(chr, positions);
    }
    int minLength = 0;
    if (PreferencesManager.getPreferences().getAsBoolean(SAM_HIDE_SMALL_INDEL)) {
        minLength = PreferencesManager.getPreferences().getAsInt(SAM_SMALL_INDEL_BP_THRESHOLD);
    }
   for (Alignment a : alignments) {
        AlignmentBlock[] blocks = a.getInsertions();
        if (blocks != null) {
            for (AlignmentBlock block : blocks) {

                if (block.getBases().length < minLength) continue;

                Integer key = block.getStart();
                InsertionMarker insertionMarker = insertionMap.get(key);
                if (insertionMarker == null) {
                    insertionMarker = new InsertionMarker(block.getStart(), block.getLength());
                    insertionMap.put(key, insertionMarker);
                    positions.add(block.getStart());
                } else {
                    insertionMarker.size = Math.max(insertionMarker.size, block.getLength());
                }
            }
        }
    }
    positions.addAll(insertionMap.keySet());
    positions.sort((o1, o2) -> o1 - o2);
}
```

```
if(insertionMap == null) {
    if(positions == null) {
    if (PreferencesManager.getPreferences().getAsBoolean(SAM_HIDE_SMALL_INDEL)) {
   for (Alignment a : alignments) {
        if (blocks != null) {
            for (AlignmentBlock block : blocks) {
                if (block.getBases().length < minLength) continue;
                if (insertionMarker == null) {
                } else {
```

```
public synchronized void processAlignments(String chr, List<Alignment> alignments) {

    Genome genome = GenomeManager.getInstance().getCurrentGenome();
    chr = genome == null ? chr : genome.getCanonicalChrName(chr);

    Map<Integer, InsertionMarker> insertionMap = insertionMaps.get(chr);
    if(insertionMap == null) {
        insertionMap =  Collections.synchronizedMap(new HashMap<>());
        insertionMaps.put(chr, insertionMap);
    }
    List<Integer> positions = positionsMap.get(chr);
    if(positions == null) {
        positions = new ArrayList<>();
        positionsMap.put(chr, positions);
    }
    int minLength = 0;
    if (PreferencesManager.getPreferences().getAsBoolean(SAM_HIDE_SMALL_INDEL)) {
        minLength = PreferencesManager.getPreferences().getAsInt(SAM_SMALL_INDEL_BP_THRESHOLD);
    }
    for (Alignment a : alignments) {
        AlignmentBlock[] blocks = a.getInsertions();
        if (blocks != null) {
            for (AlignmentBlock block : blocks) {

                if (block.getBases().length < minLength) continue;

                Integer key = block.getStart();
                InsertionMarker insertionMarker = insertionMap.get(key);
                if (insertionMarker == null) {
                    insertionMarker = new InsertionMarker(block.getStart(), block.getLength());
                    insertionMap.put(key, insertionMarker);
                    positions.add(block.getStart());
                } else {
                    insertionMarker.size = Math.max(insertionMarker.size, block.getLength());
                }
            }
        }
    }
    positions.addAll(insertionMap.keySet());
    positions.sort((o1, o2) -> o1 - o2);
}
```

Becoming a code speed-reader (4 techniques)

TECHNIQUE #4: Start reading from the outputs.

TECHNIQUE #5: Find the frequent words.

TECHNIQUE #6: Filter on control flow.

TECHNIQUE #7: Scan for the main action.

10 TECHNIQUES TO UNDERSTAND CODE YOU DON'T KNOW

OUTLINE:

- Exploring the code (3 techniques)
- Becoming a code speed-reader (4 techniques)
- Understanding code in details (3 techniques)

Understanding code in detail

(3 techniques)

TECHNIQUE #8

Decouple the code.

```
class Order
{
public:
    Order(double price, Country const& originCountry, Country const& destinationCountry);

    double getTotalPrice() const;
    // ...
    void process();
    // ...

private:
    void processTaxes();

    // ...
    double price_;
    double totalPrice_;
    // ...
    Country originCountry_;
    Country destinationCountry_;
    // ...
};
```

```
void Order::process()
{
    // ...
    processTaxes();
    // ...
}
```

```
void Order::processTaxes()
{
   double taxValue = price_ * getTaxRate(destinationCountry_);
   double internationalTaxCredit = getInternatinalTaxCreditValue(price_, originCountry_,
                                                                 destinationCountry_);
   double taxCut = getTaxCutRate(price_) * price_;
   totalPrice_ = price_ + taxValue;
   double taxReduction = std::min(internationalTaxCredit + taxCut, totalPrice_);
   totalPrice_ -= taxReduction;
}

class Order
{
public:
    // ...

private:
    void processTaxes();
    // ...
    double price_;
    double totalPrice_;
    // ...
    Country originCountry_;
    Country destinationCountry_;
    // ...
};
```

The compiler helps
locate data member accesses

```
void processTaxes(Order& order)
{
   double taxValue = order.getPrice() * getTaxRate(order.getDestinationCountry());
   double internationalTaxCredit = getInternatinalTaxCreditValue(order.getPrice(),
                                   order.getOriginCountry(), order.getDestinationCountry());
   double taxCut = getTaxCutRate(order.getPrice()) * order.getPrice();
   order.setTotalPrice(order.getPrice() + taxValue);
   double taxReduction = std::min(internationalTaxCredit + taxCut, order.getTotalPrice());
   order.setTotalPrice(order.getTotalPrice() - taxReduction);
}

class Order
{
public:
    // ...

private:
    // ...
    double price_;
    double totalPrice_;
    // ...
    Country originCountry_;
    Country destinationCountry_;
    // ...
};

void processTaxes(Order& order);
```

Getters
- getPrice()
- getOriginCountry()
- getDestinationCountry()
- getTotalPrice()

Setters
- setTotalPrice()

```
double processTaxes(double price, Country const& originCountry,
                                  Country const& destinationCountry, double totalPrice)
{
    double taxValue = price * getTaxRate(destinationCountry);
    double internationalTaxCredit = getInternatinalTaxCreditValue(price, originCountry,
                                                                  destinationCountry);
    double taxCut = getTaxCutRate(price) * price;
    totalPrice = price + taxValue;
    double taxReduction = std::min(internationalTaxCredit + taxCut, totalPrice);
    return totalPrice - taxReduction;
}
```

```
double priceAfterTaxes(double price, Country const& originCountry, Country const&
                                                                    destinationCountry)
{
    double taxValue = price * getTaxRate(destinationCountry);
    double internationalTaxCredit = getInternatinalTaxCreditValue(price, originCountry,
                                                                    destinationCountry);
    double taxCut = getTaxCutRate(price) * price;
    double totalPrice = price + taxValue;
    double taxReduction = std::min(internationalTaxCredit + taxCut, totalPrice);
    return totalPrice - taxReduction;
}
```

```
void Order::process()
{
    // ...
    totalPrice_ = priceAfterTaxes(price_, originCountry_, destinationCountry_);
}
```

```
double priceAfterTaxes(double price, Country const& originCountry, Country const&
                                                                    destinationCountry)
{
    double taxValue = price * getTaxRate(destinationCountry);
    double internationalTaxCredit = getInternatinalTaxCreditValue(price, originCountry,
                                                                    destinationCountry);
    double taxCut = getTaxCutRate(price) * price;
    double totalPrice = price + taxValue;
    double taxReduction = std::min(internationalTaxCredit + taxCut, totalPrice);
    return totalPrice - taxReduction;
}
```

```
void Order::process()
{
    // ...
    priceAfterTaxes_ = priceAfterTaxes(price_, originCountry_, destinationCountry_);
}
```

- Scratch refactoring: throw it away

TECHNIQUE #9

Identify practice-functions.

What is a practice-function?

- Complex code
- No dependencies

TECHNIQUE #10

Team up.

- A fresh look
- Higher returns
- Rubber-ducking
- Sum of knowledge
- Sustained focus

Understanding code in detail

TECHNIQUE #8: Decouple the code.

TECHNIQUE #9: Identify practice-functions.

TECHNIQUE #10: Team up.

10 TECHNIQUES TO UNDERSTAND CODE YOU DON'T KNOW

Exploring the code

- #1: Know your I/O frameworks.
- #2: Find a stronghold.
- #3: Analyse call stacks.

Becoming a code speed-reader

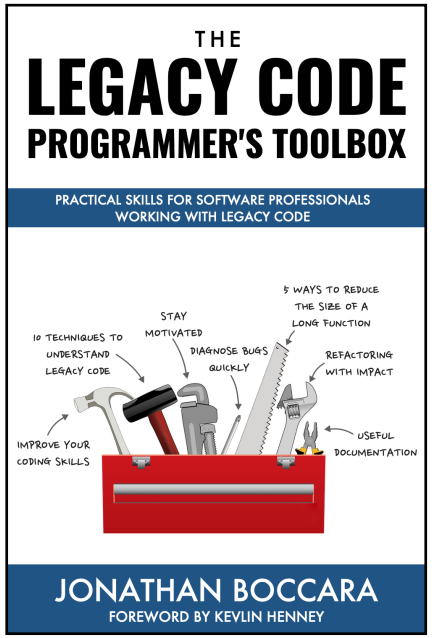
- #4: Start reading from the outputs.
- #5: Count words.
- #6: Filter on control flow.
- #7: Scan for the main action.

Understanding code in details

- #8: Decouple the code.
- #9: Identify practice-functions.
- #10: Team up.

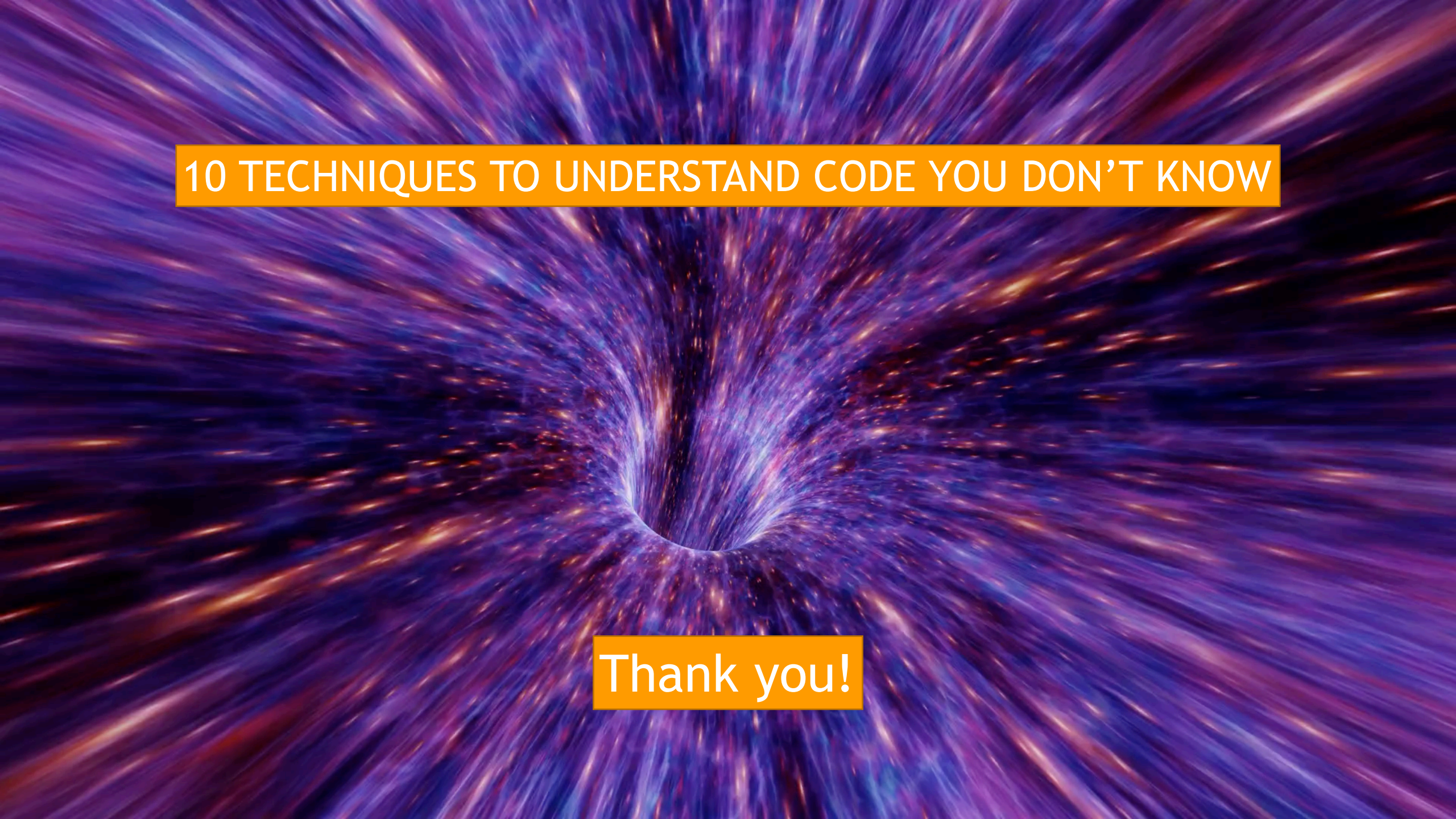
@JoBoccara

ABOUT WORKING WITH EXISTING CODE



leanpub.com/legacycode

Book signing tomorrow
Conservatory @lunchtime


10 TECHNIQUES TO UNDERSTAND CODE YOU DON'T KNOW
Thank you!

10 TECHNIQUES TO UNDERSTAND CODE YOU DON'T KNOW

Exploring the code

- #1: Know your I/O frameworks.
- #2: Find a stronghold.
- #3: Analyse call stacks.

Becoming a code speed-reader

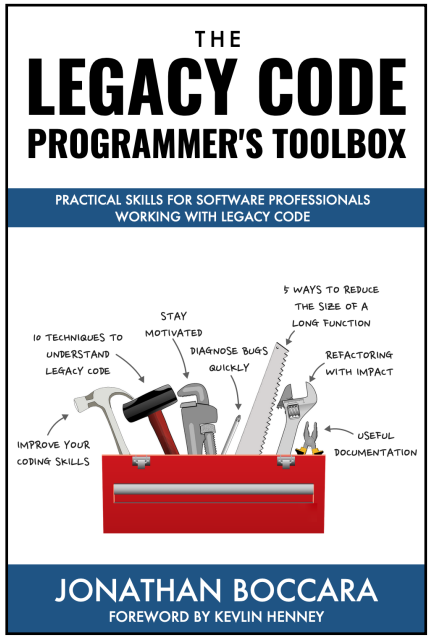
- #4: Start reading from the outputs.
- #5: Count words.
- #6: Filter on control flow.
- #7: Scan for the main action.

Understanding code in details

- #8: Decouple the code.
- #9: Identify practice-functions.
- #10: Team up.

@JoBoccara

ABOUT WORKING WITH EXISTING CODE



leanpub.com/legacycode

Book signing tomorrow
Conservatory @lunchtime