Diving into the KiCad source code

creating new tools in KiCad

Maciej Sumiński

FOSDEM @ Brussels
5th February 2017

Agenda

- Building KiCad
- Tool Framework
- Useful classes
- Example
- Submitting patches
- Documentation & support

Building KiCad

Linux

[install dependencies]

```
git clone https://git.launchpad.net/kicad

mkdir kicad/build

cd kicad/build

cmake ..

make
```

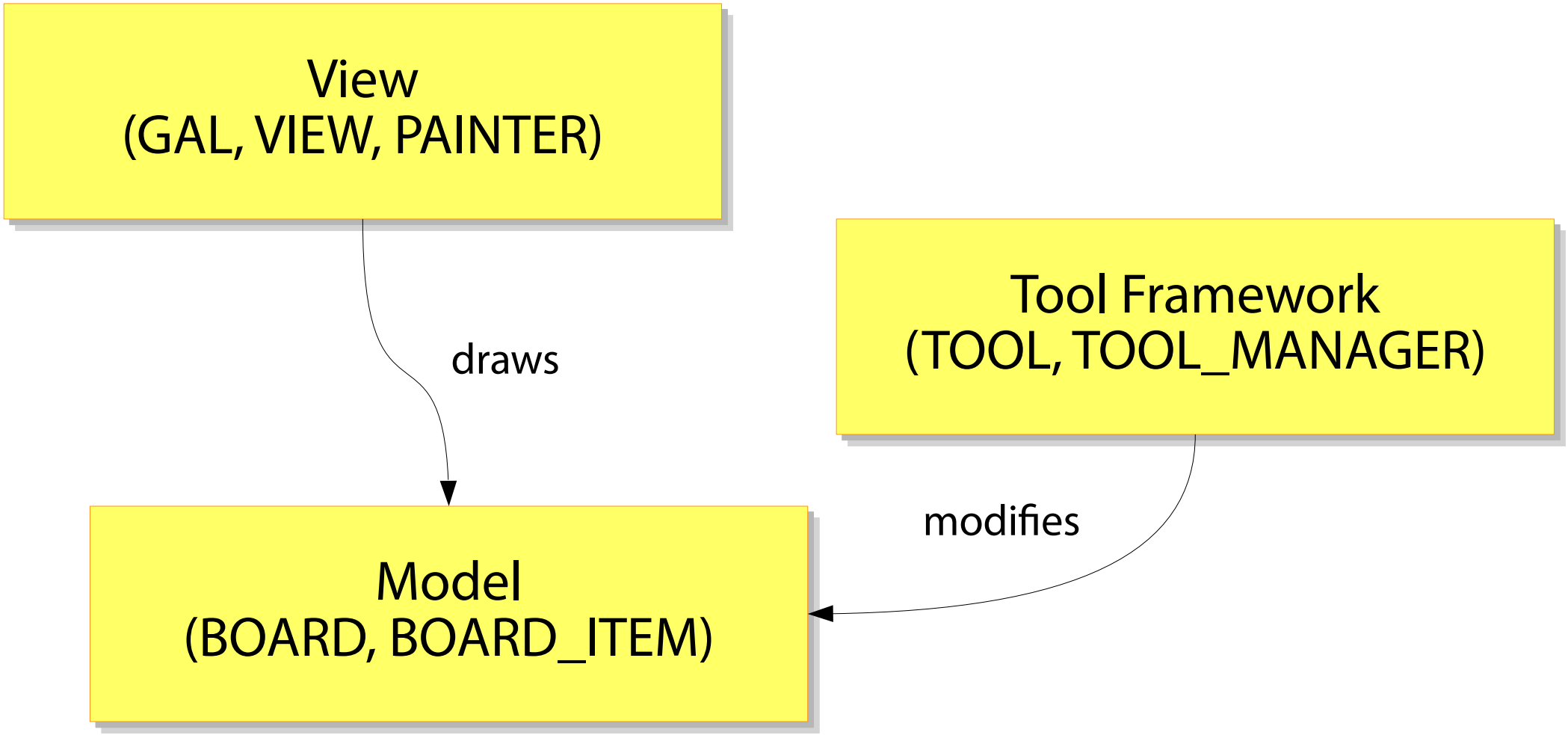
Building KiCad

Windows

[install MSYS2 & dependencies]

```
git clone https://github.com/Alexpux/MINGW-packages

cd MinGW-packages/mingw-w64-kicad-git

makepkg-mingw -is
```

Tool Framework



Tool Framework

```
class NEW_TOOL : public PCB_TOOL
{
public:
    NEW_TOOL();
    ~NEW_TOOL();
    void Reset( RESET_REASON aReason );
    bool Init();
    void SetTransitions();

private:
    int eventHandler( const TOOL_EVENT& aEvent );
};
```

```
NEW_TOOL::NEW_TOOL() :
    PCB_TOOL( “pcbnew.ToolName” )
{
    // initialize variables
}
```

```
NEW_TOOL::~NEW_TOOL()
{
    // deinitialize variables
}
```

Tool Framework

```
class NEW_TOOL : public PCB_TOOL
{
public:
    NEW_TOOL();
    ~NEW_TOOL();
    void Reset( RESET_REASON aReason );
    bool Init();
    void SetTransitions();

private:
    int eventHandler( const TOOL_EVENT& aEvent );
};
```

```
NEW_TOOL::Reset( ... )
{
    switch( aReason )
    {
            case RUN:
                ...
            case MODEL_RELOAD:
                ...
            case GAL_SWITCH:
                ...
    }
}
```

Tool Framework

```
class NEW_TOOL : public PCB_TOOL
{
public:
    NEW_TOOL();
    ~NEW_TOOL();
    void Reset( RESET_REASON aReason );
    bool Init();
    void SetTransitions();

private:
    int eventHandler( const TOOL_EVENT& aEvent );
};
```

```
NEW_TOOL::Init()
{
    // one-time initialization
    // when other tools are also
    // initialized, e.g. adding menus
}
```

Tool Framework

```
class NEW_TOOL : public PCB_TOOL
{
public:
    NEW_TOOL();
    ~NEW_TOOL();
    void Reset( RESET_REASON aReason );
    bool Init();
    void SetTransitions();

private:
    int eventHandler( const TOOL_EVENT& aEvent );
};
```

```
NEW_TOOL::SetTransitions()
{
    // set event handlers
    Go( &NEW_TOOL::eventHandler, eventA );
}
```

Tool Framework

```
class NEW_TOOL : public PCB_TOOL
{
public:
    NEW_TOOL();
    ~NEW_TOOL();
    void Reset( RESET_REASON aReason );
    bool Init();
    void SetTransitions();

private:
    int eventHandler( const TOOL_EVENT& aEvent );
};
```

```
NEW_TOOL::eventHandler( ... )
{
    // single action handler
    // execute an action and finish
    selectedItem->SetPosition( wxPoint( 0, 0 ) );
}
```

Tool Framework

```
class NEW_TOOL : public PCB_TOOL
{
public:
    NEW_TOOL();
    ~NEW_TOOL();
    void Reset( RESET_REASON aReason );
    bool Init();
    void SetTransitions();

private:
    int eventHandler( const TOOL_EVENT& aE
};
```

```
NEW_TOOL::eventHandler( ... )
{
    // single action handler
    // execute an action and finish
    selectedItem->SetPosition( wxPoint( 0, 0 ) );
}
```

```
NEW_TOOL::eventHandler( ... )
{
    // interactive tool
    while( OPT_TOOL_EVENT evt = Wait() )
    {
        if( evt->IsClick( BUT_LEFT )
            ...
        else if( evt->IsDrag( BUT_LEFT )
            ...
    }
}
```

Tool Framework

- Add the new file to pcbnew/CMakeLists.txt
- Add to pcbnew/tools/tools_common.cpp:

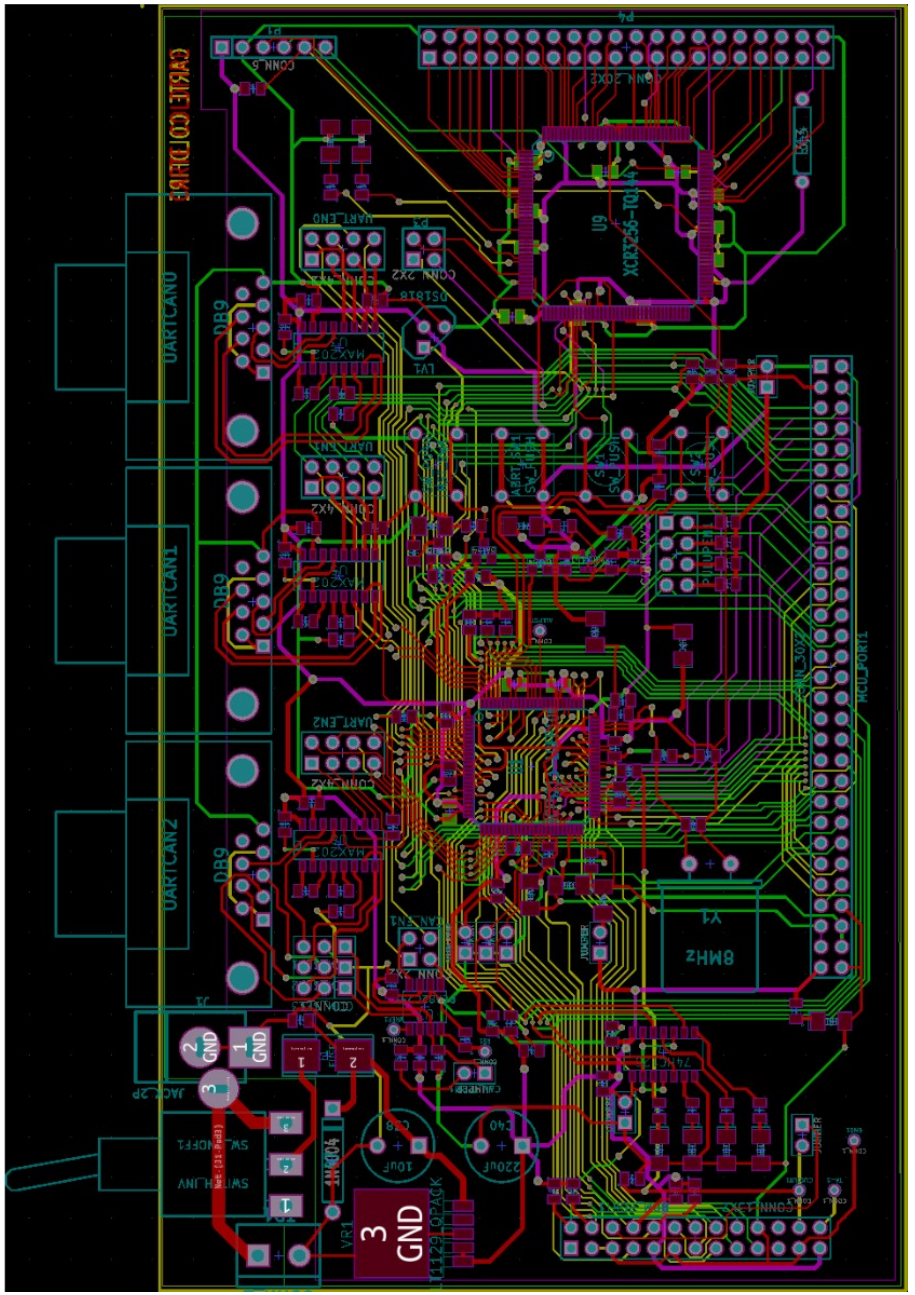
```
#include <tools/new_tool.h>

void registerAllTools( TOOL_MANAGER* aToolManager )
{
    …
    aToolManager->RegisterTool( new NEW_TOOL );
    …
}
```

Useful classes: model

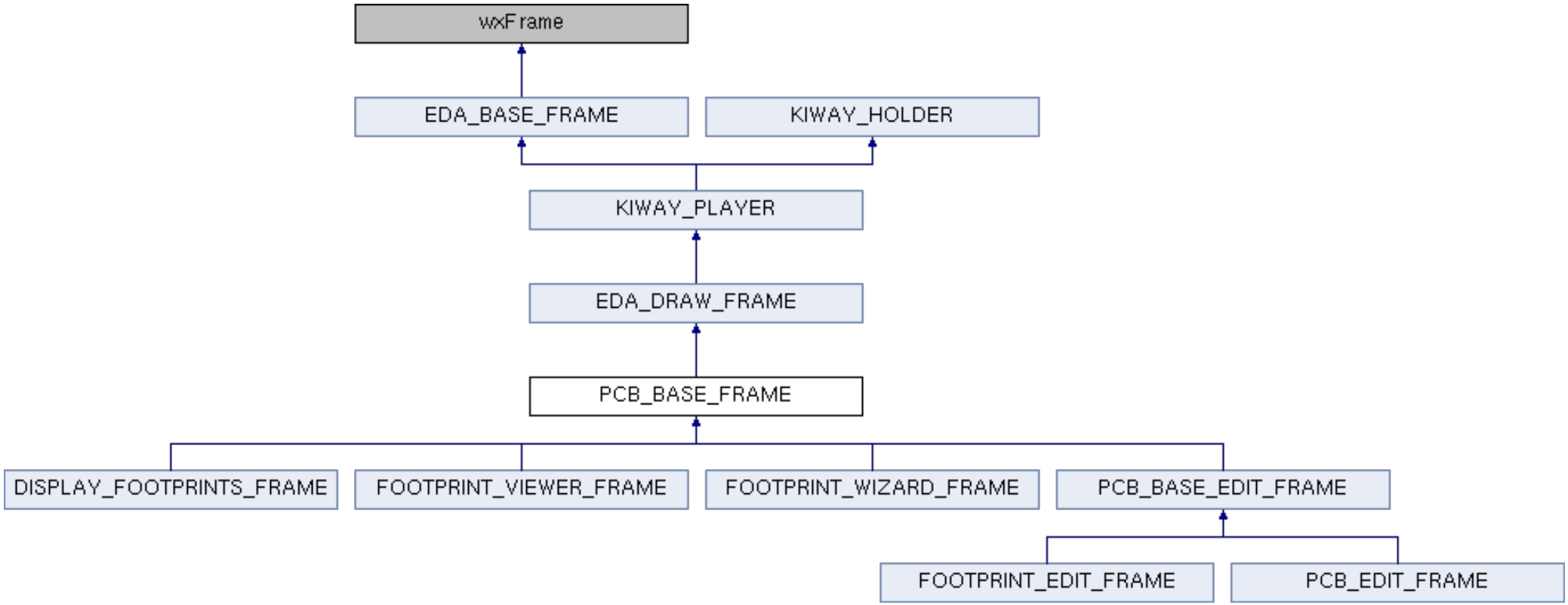
EDA_ITEM

BOARD_ITEM

- BOARD_CONNECTED_ITEM
  - D_PAD
  - TRACK
  - VIA
  - ZONE_CONTAINER
- BOARD_ITEM_CONTAINER
  - BOARD
  - MODULE
- DRAWSEGMENT
- NETINFO_ITEM
- TEXTE_MODULE
- TEXTE_PCB

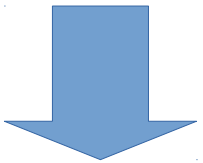


Useful classes: windows



Useful classes: introducing changes

```
BOARD_COMMIT commit( frame );
commit.Add( new TRACK( … ) )
commit.Modify( module );      // saves the state
// perform module modifications
commit.Remove( drawing );
commit.Push( “Updated board” );
```



create undo entry
update model, view, ratsnest

Useful classes: actions

```
TOOL_ACTION flip( "pcbnew.InteractiveEdit.flip",
    AS_GLOBAL, ‘F’,
    _( "Flip" ), _( "Flips selected item(s)" ),
    swap_layer_xpm );
```

Useful classes: managing tools

TOOL_MANAGER

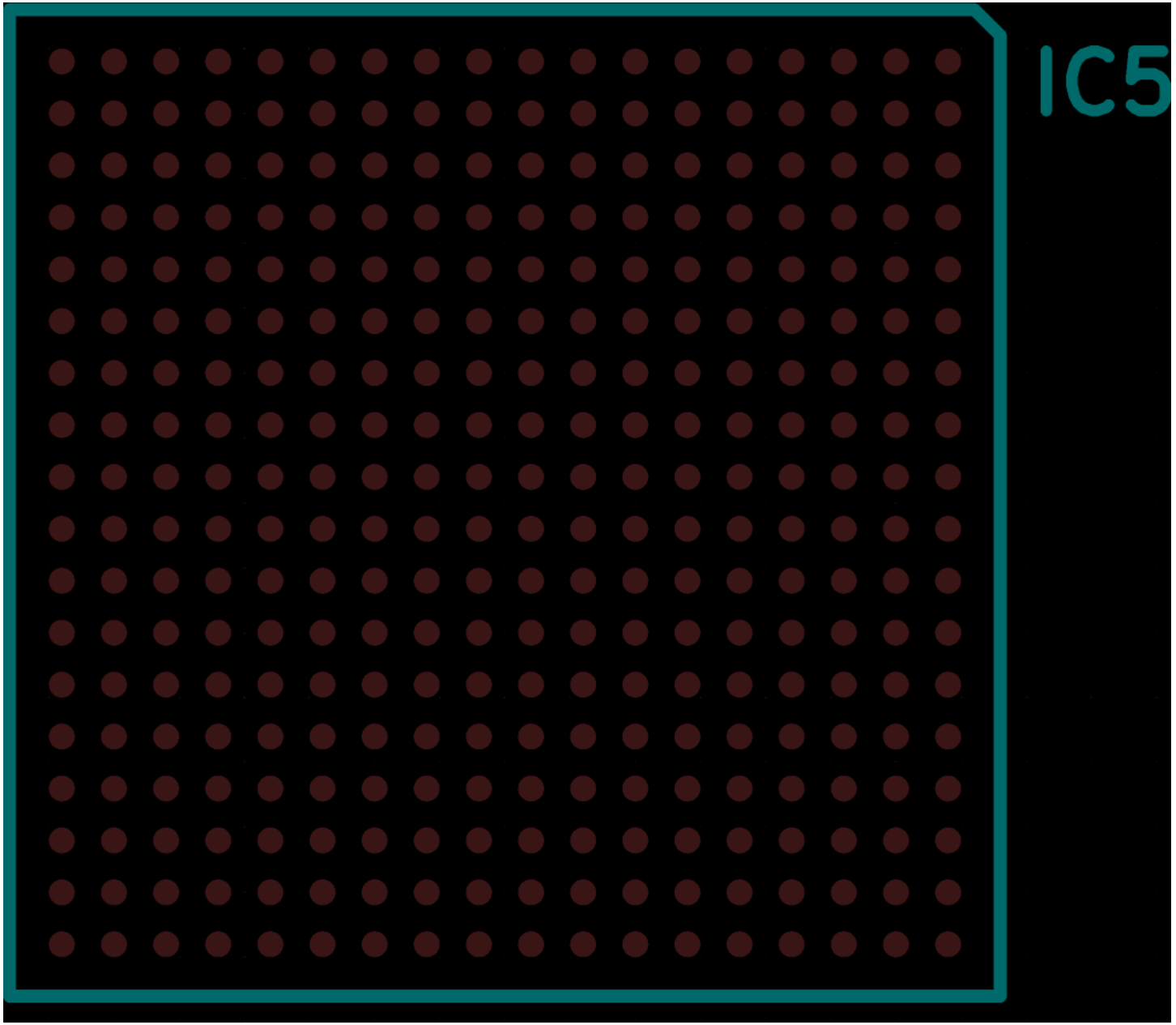
- InvokeTool()
- GetTool()
- RunAction()

interact with other tools

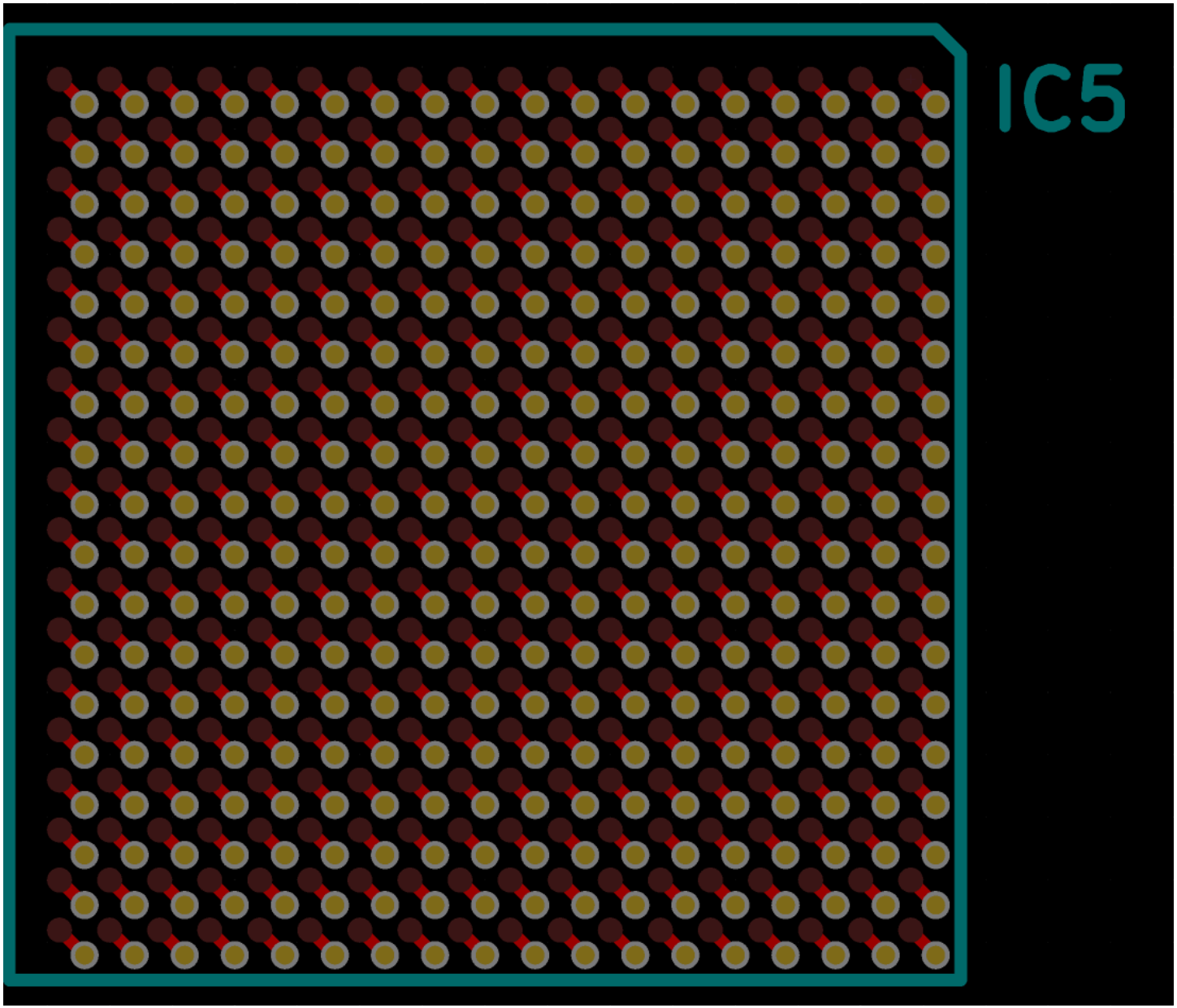
- GetView()
- GetModel()
- GetEditFrame()

access essential objects

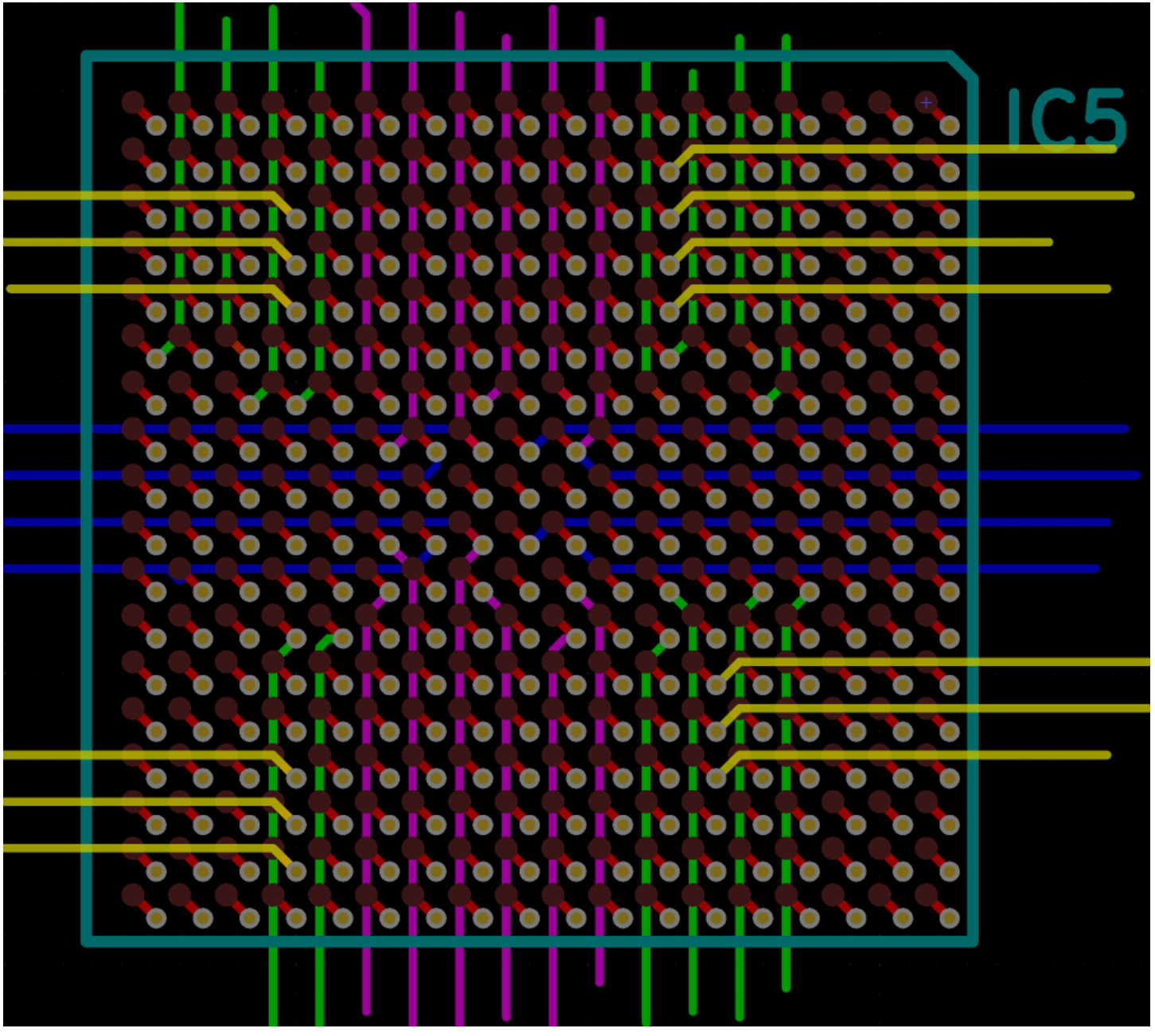
Example: fanout tool



Example: fanout tool



Example: fanout tool



/* code & short demo */

```
git clone -b fanout \
https://git.launchpad.net/~orsonmmz/kicad
```

More examples...

kicad-src/pcbnew/tools:

- PLACEMENT_TOOL
- ZOOM_TOOL
- PCB_EDITOR_CONTROL

Submitting patches

- Code formatting
- Single patches: e-mail on developer’s mailing list
- Large batch: merge request on Launchpad

Documentation & support

- Official KiCad webpage

  http://kicad-pcb.org/contribute/developers/

- Tool Framework tutorial

  http://kicad.readthedocs.io/en/latest/Documentation/development/tool-framework

- KiCad developer mailing list

  https://launchpad.net/~kicad-developers

- #kicad@freenode.net

Questions?