ThreadWeaver in KDE Frameworks 5

Demos, New Features, Performance

Mirko Boehm <mirko@kde.org>

Akademy 2013

Agenda

- introduction to ThreadWeaver
- key concepts: job aggregates, policies, queues
- API changes for Frameworks 5
- performance considerations, benchmarks
- outlook and new feature quizz

CWD: ~
miroslav@silberpfeil: >whoami

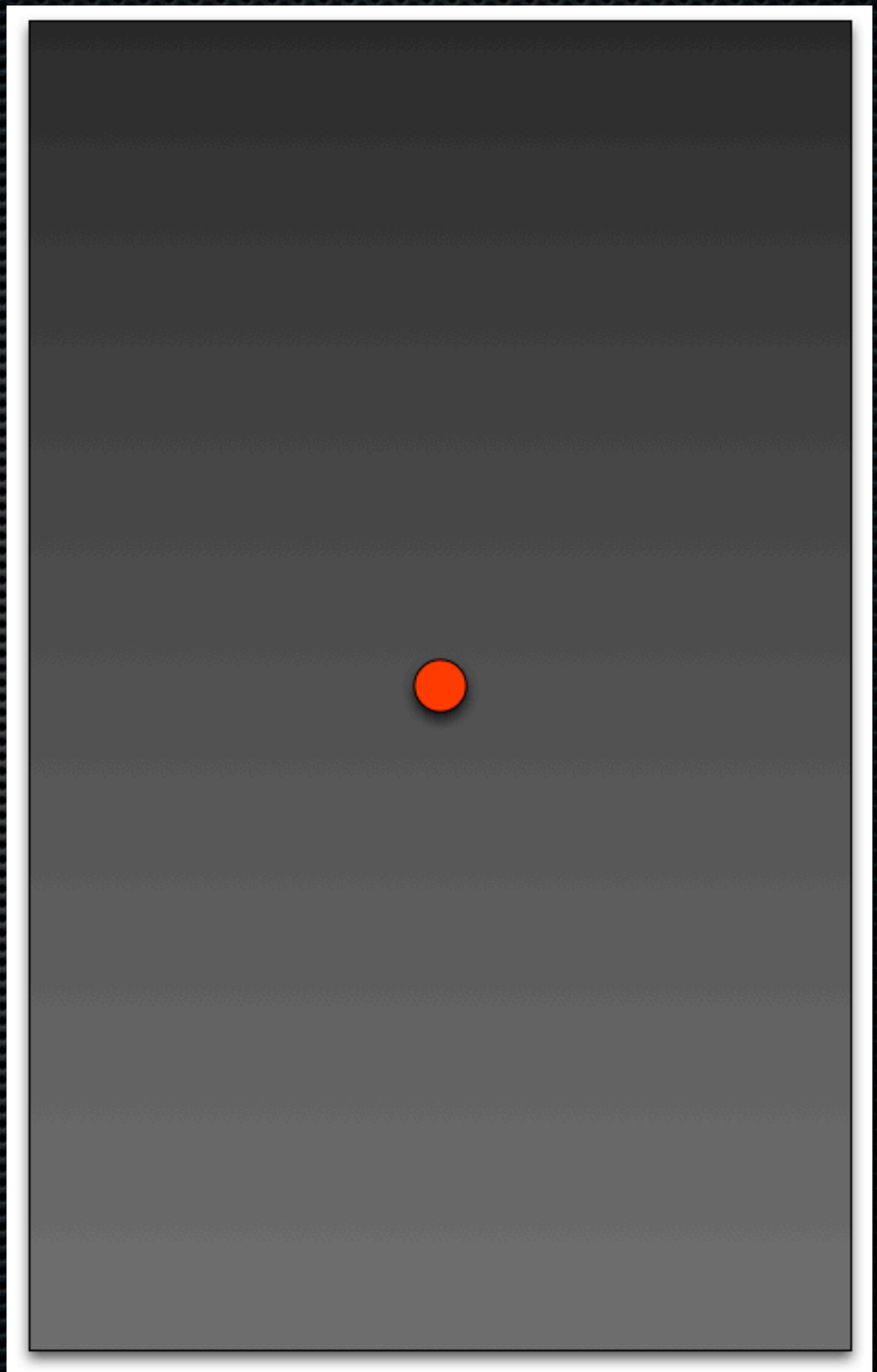
```
- KDE Contributor since 1997:
-- hacking (kdecore, kdepim, applications)
-- board member 1999 to 2006
-- Desktop Summit 2011
- FSFE Team Germany
- researching Free Software and Intellectual Property
  issues at TU Berlin
- European Representative, Open Invention Network
- Founder/CEO,Endocode AG
- married, two kids, lives in Berlin
CWD: ~
miroslav@silberpfeil: >
```

Agenda

- introduction to ThreadWeaver
- key concepts: job aggregates, policies, queues
- API changes for Frameworks 5
- performance considerations, benchmarks
- outlook and new feature quizz

Hello World

Demo time!

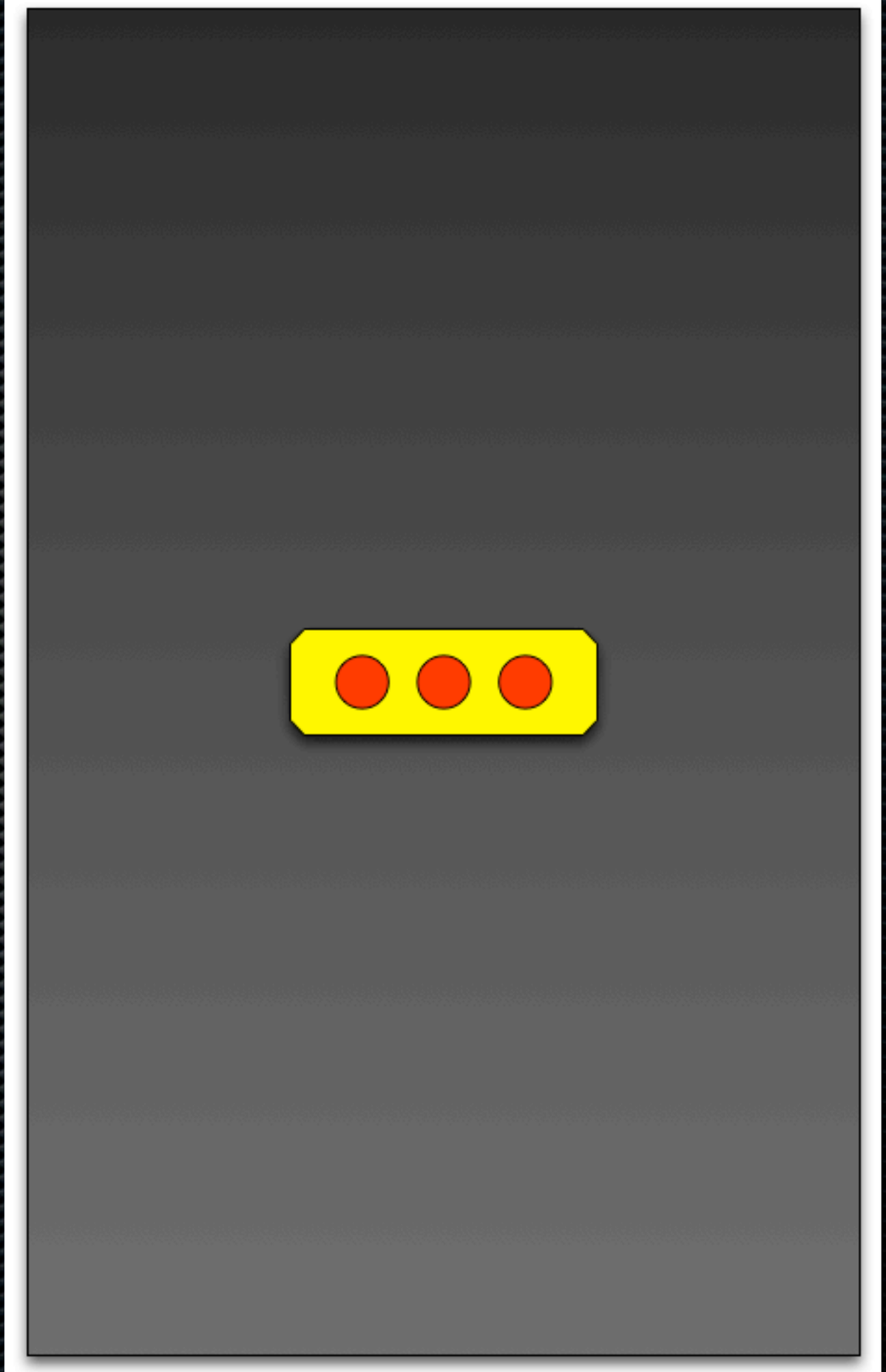


Hello World demo(s)

- global ThreadWeaver pool
- job memory management
  - shared pointers
  - raw pointers, jobs as stack variables
- Lambda jobs

Sequence Demo

Demo time!



Sequence Demo

- job aggregates: collections, sequences
- job error reporting
  - sequences abort on failure
- signals and slots to communicate
- sharing data between jobs

Image Viewer

Demo time!

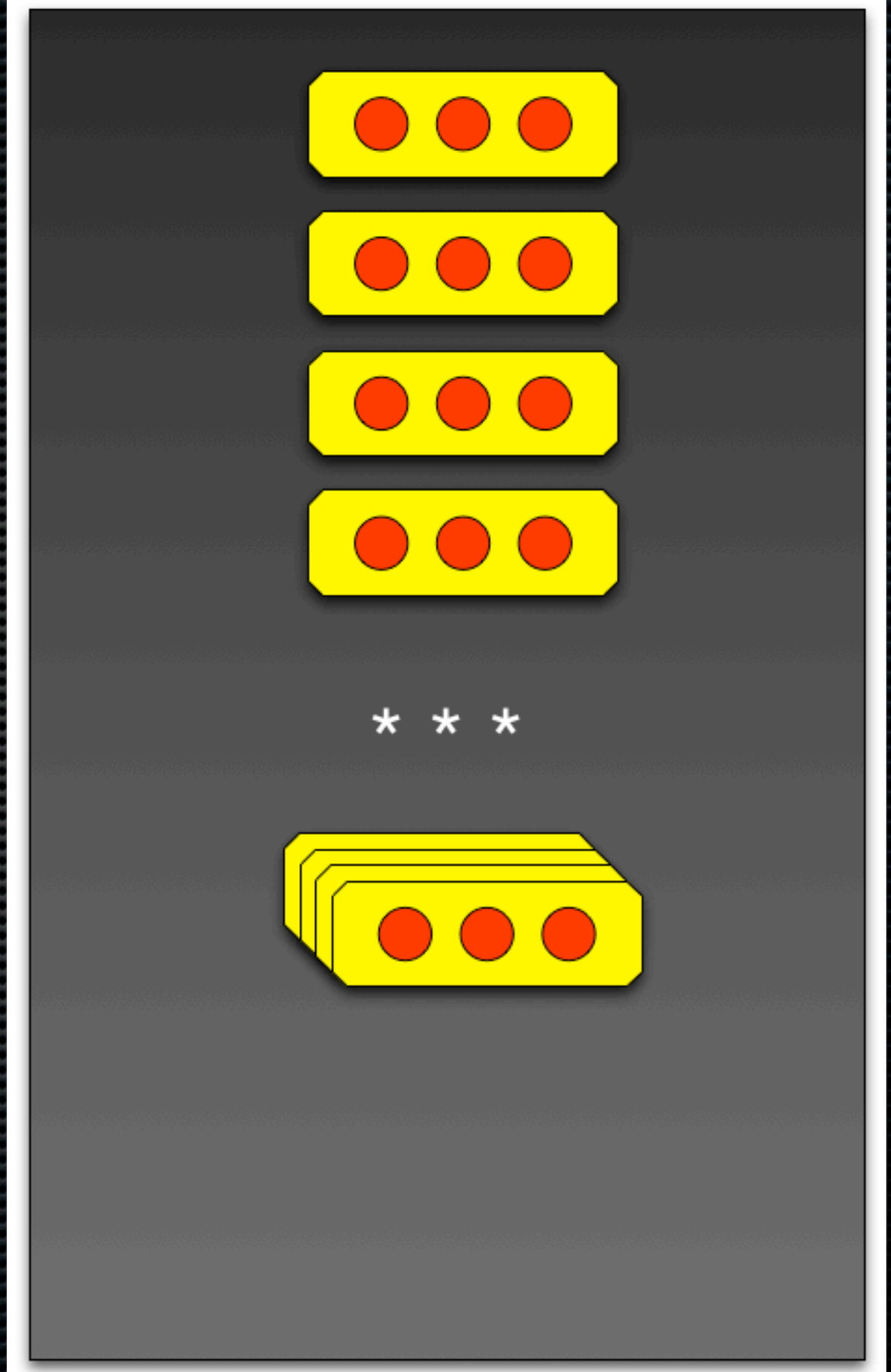


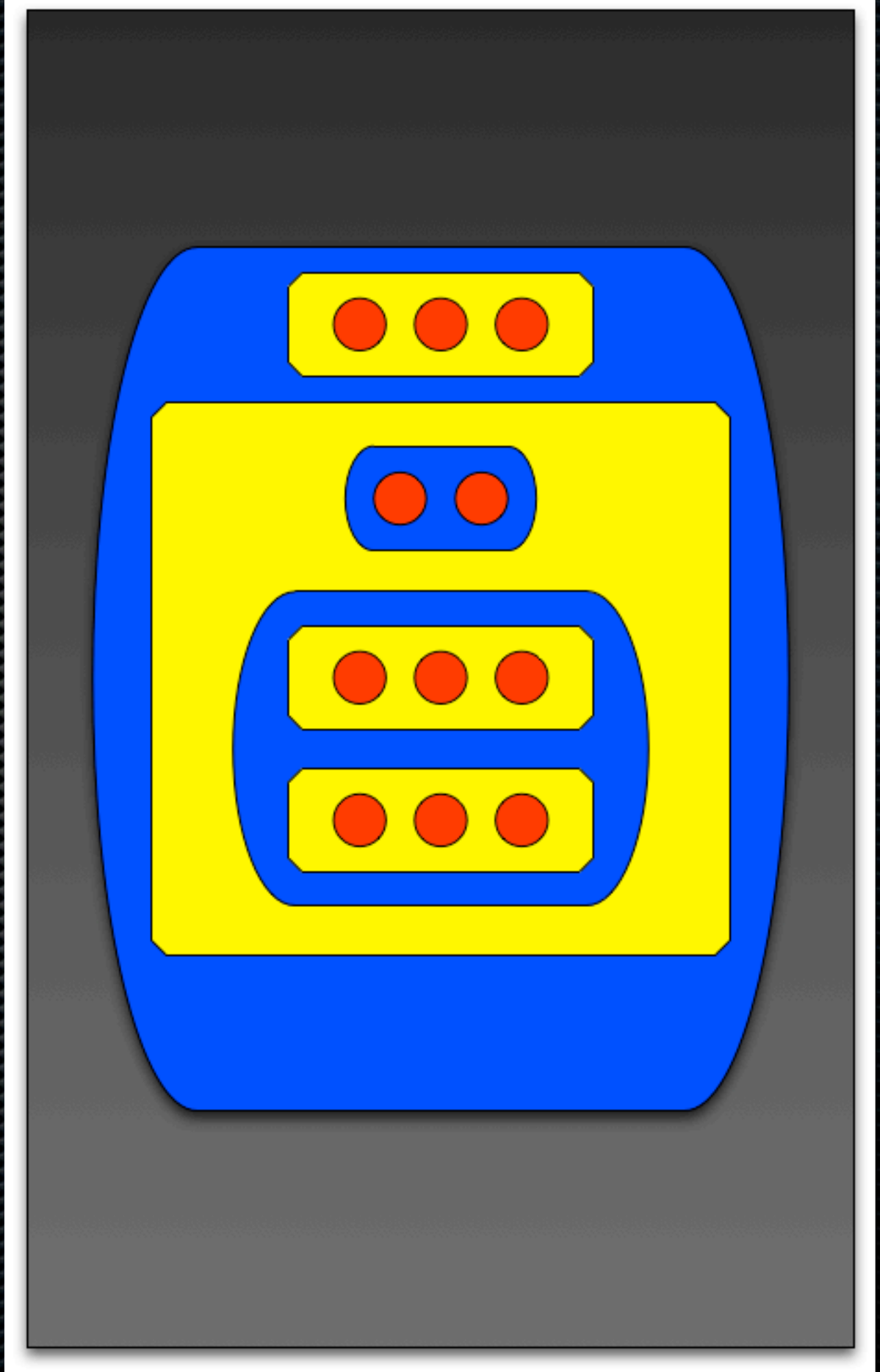
Image Viewer

- queue policies
  - resource restrictions
    - dependencies
  - job priorities
  - worker thread count, individual queue instances

Outlook

Use job aggregates to model execution flow graph.

(Or: How to draw a sad robot.)



Agenda

- introduction to ThreadWeaver
- key concepts: job aggregates, policies, queues
- API changes for Frameworks 5
- performance considerations, benchmarks
- outlook and new feature quizz

Jobs

- units of queueing and execution
- shared pointers for memory management
  - raw pointers for stack variables
- success status,
- cancellation requests
- priorities as a hint to the scheduler

Queue Policies

- decide whether or not a job can be executed
- are assigned to jobs
- may be shared between jobs or queues
- built-ins:
  - resource restriction policy, dependency policy
- custom policies can be implemented

Queues

- manage worker threads
- global instance or individual instances
- suspend/resume/finish
- signaling

Philosophy

- scheduling vs. programmed concurrency
- critical path modeling
- minimalism (jobs, queueing methods, concepts)
- simplicity

Agenda

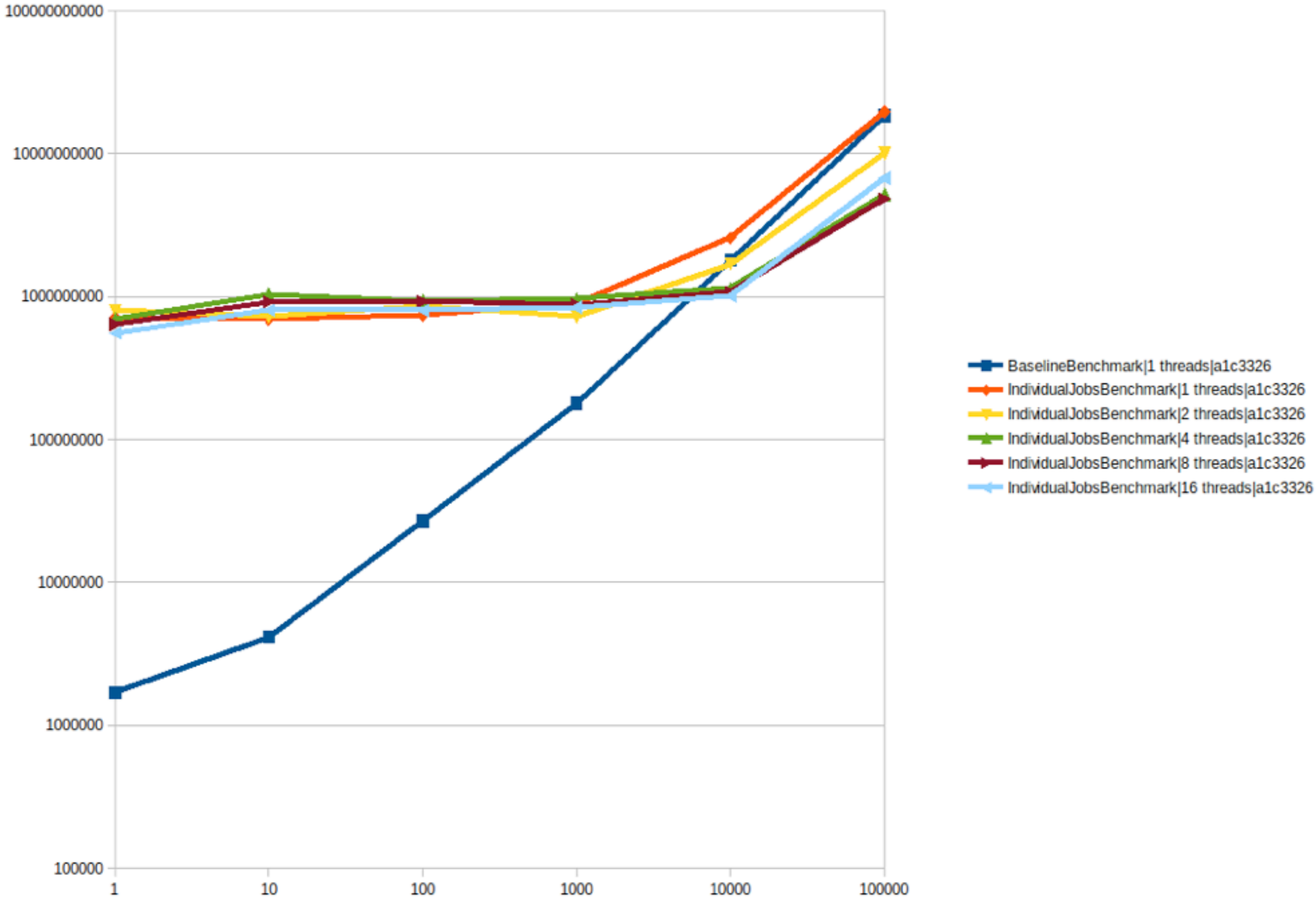
- introduction to ThreadWeaver
- key concepts: job aggregates, policies, queues
- API changes for Frameworks 5
- performance considerations, benchmarks
- outlook and new feature quizz

API changes

- few, but important
- shared pointers for jobs
- global instance child of QApplication
- public Queue class
- ExecuteWrapper
- all *RunHelper* classes removed

Agenda

- introduction to ThreadWeaver
- key concepts: job aggregates, policies, queues
- API changes for Frameworks 5
- performance considerations, benchmarks
- outlook and new feature quizz


100000000000
10000000000
1000000000
100000000
10000000
1000000
100000
1
10
100
1000
10000
100000
BaselineBenchmark|1 threads|a1c3326
IndividualJobsBenchmark|1 threads|a1c3326
IndividualJobsBenchmark|2 threads|a1c3326
IndividualJobsBenchmark|4 threads|a1c3326
IndividualJobsBenchmark|8 threads|a1c3326
IndividualJobsBenchmark|16 threads|a1c3326

Agenda

- introduction to ThreadWeaver
- key concepts: job aggregates, policies, queues
- API changes for Frameworks 5
- performance considerations, benchmarks
- outlook and new feature quizz

Feature quizz

- progress tracking
- minijobs
- embed QRunnables
- Job::execUntilFinished(QEventLoop/QDialog/qApp)
- AnyOfCollection
- UI elements/visualizations

Questions?

References

- demos:

  github.com/mirkoboehm/ThreadWeaverDemos

- benchmarks:

  github.com/mirkoboehm/QBenchmarkParser

- code:

  kdelibs frameworks branch, tier1/threadweaver