The scripts

Some myths

• CONTEXT looks a lot like plain TEX and expects users to program macros.

• CONTEXT depends on RUBY.

The truth

• On the average users don’t have to program. Configuring is not programming.

• As TEX lacks commandline handling and job control, helpers are provided.

• Of course users can still program a lot, but not all need that.

• Of course users can directly run CONTEXT, but why should they.

A few facts

• The CONTEXT distribution provides a sort of ecosystem.

• In MKII indeed we use RUBY for some job control.

• But in MKIV all is (of course) done in LUA.

• Two scripts play an important role: mtxrun and context.
The scripts

The 'mtxrun' script

- Locates and runs scripts, has a lot of helpers preloaded.
- It is in fact my LUA runner on top the TEXLUA.
- It knows about files and the environment we run in.
- It has some features that makes it easier to integrate in services.
- This way we don't need stubs (and avoid potential conflicts in name).

The 'context' script

- It runs CONTEXT and keeps track of how many runs are needed.
- Contrary to its MKII ancestor it is not needed for index sorting etc.
- It has a few extensions that are loaded on demand: extras
Breskens 2012

The scripts

A regular run
--run filename

Running from an editor
--autopdf filename

Running from a service
mtxrun --path=somepath --script context filename
Controlling the rendering
--usemodule=list
--environment=list
--mode=list
--arguments=list
--path=list

Controlling with ctx files
--ctx=name

Also in preamble
<?context-directive job ctxfile m4all.ctx ?>
<?xml version='1.0' standalone='yes'?>
<ctx:job>
  <ctx:message>EPUB Formatter</ctx:message>
  <ctx:flags>
    <ctx:flag>purge</ctx:flag>
    <ctx:flag>global</ctx:flag>
  </ctx:flags>
  <ctx:process>
    <ctx:resources>
      <ctx:module>epub-01</ctx:module>
    </ctx:resources>
  </ctx:process>
</ctx:job>
Breskens 2012

--result=name

When imposition is needed

--arrange

Cleanup after runs

--batchmode

--purge(all)

--purgeresult

The scripts
Sometimes faster (in services) --once
Normally automatically done --make --generate --touch
Seldom used --interface --randomseed=number
Information about extra control

--trackers
--directives
--showlogcategories
--version

Controlling the machinery

--trackers=list
--directives=list
--silent=list
--noconsole
--nostatistics
When no local file is used:

```
--global
--nofile
```

When the automatic recognition doesn't work:

```
--forcexml
--forcecld
--forcelua
--forcemp
```
Only handy for development (or me)

--profile
--timing

Forget about these

--paranoid
--update

Some hidden treasures

--extras
--extra=name
Arguments can be prefixed:

- environment:
- relative:
- auto:
- locate:
- filename:
- pathname:
- home:
- selfautoloc:
- selfautoparent:
- selfautodir:
Recent (probably unnoticed) change

luatex

--fmt=".../tex/texmf-cache/luatex-cache/context/.../formats/cont-en"
--lua=".../tex/texmf-cache/luatex-cache/context/.../formats/cont-en.lui"
--jobname="context-the-script"
--no-parse-first-line
--c:autopdf
--c:currentrun=1
--c:fulljobname="./context-the-script.tex"
--c:input="./context-the-script.tex"
--c:kindofrun=1
"cont-yes.mkiv"

Another change

• The (runtime generated) options file is no longer there.
• For as far as possible arguments are passed directly.
• Input files are always loaded indirectly, no more stubs.
The scripts

- mtx-check.lua
- mtx-convert.lua
- mtx-epub.lua
- mtx-fonts.lua
- mtx-modules.lua
- mtx-patterns.lua
- mtx-pdf.lua

These are run like

```
mtxrun --script pdf
```
These are run like
context --extra=arrange 
[--help] ...