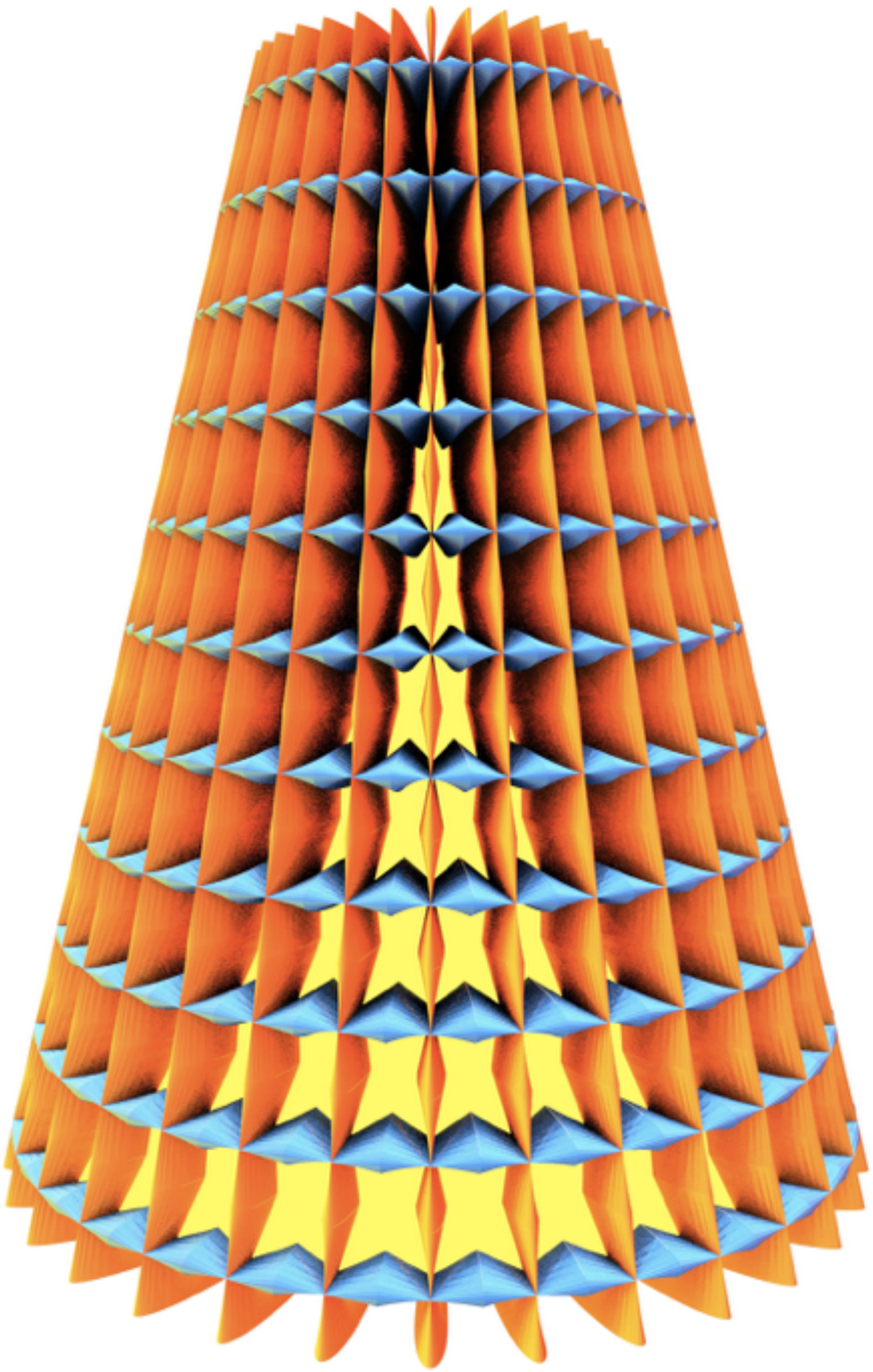


101worker

Ralf Lämmel
Software Languages Team
University of Koblenz-Landau

Acknowledgement: several team members have contributed to the development of 101worker over several generations — special thanks go to Kevin Klein.

101 recap

- 101companies — the name of the project
- 101system — the name of the sample system
- 101contributions — implementations of 101system
- 101repo — the repository with 101contributions
- 101wiki — documentation of 101contributions et al.
- **101worker — computational infrastructure on 101repo /101wiki**
- …

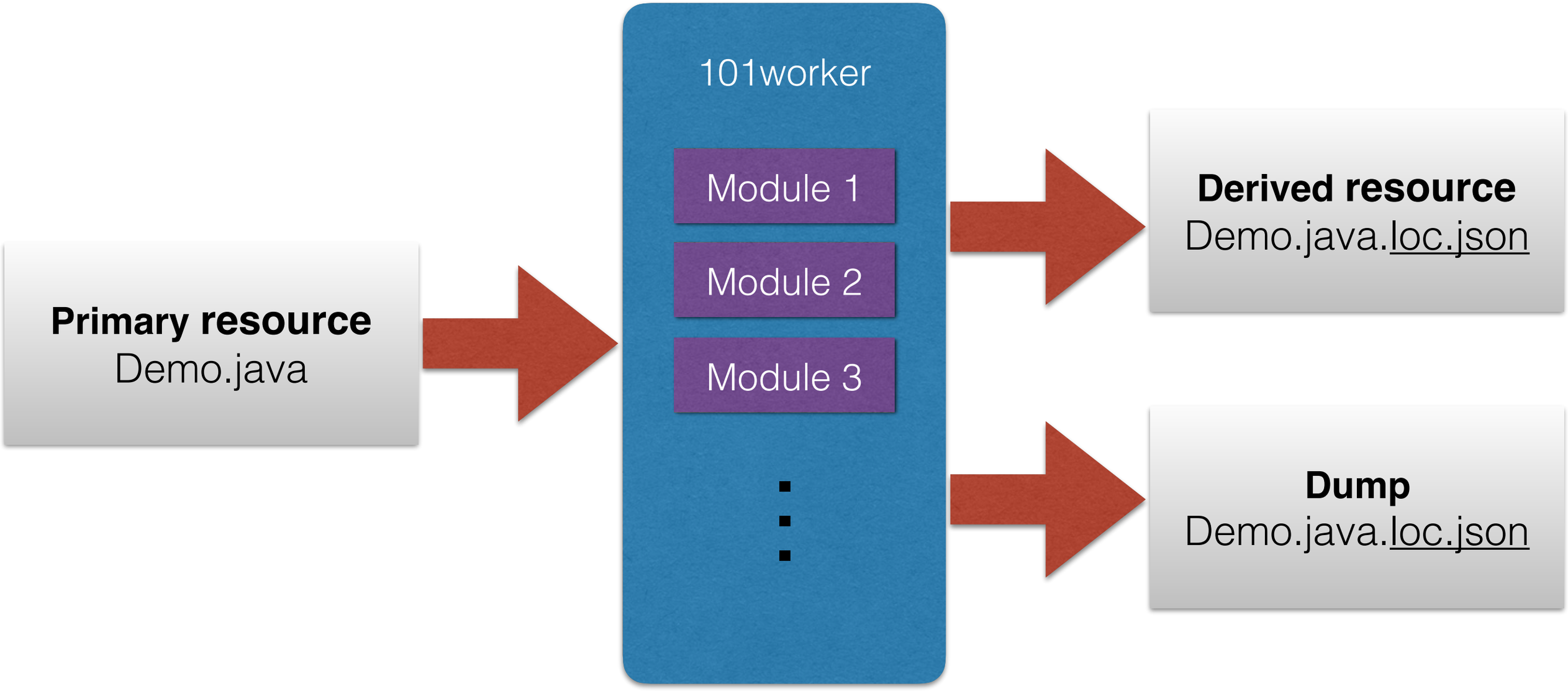
What is 101worker good for?

- 101 collects diverse artifacts:
  - Different languages, technologies, designs, features
- We may have many questions about the corpus.
  - How complex are the systems?
  - What languages or technologies are used?
  - What packages are imported with what frequency?
- 101worker uses “**modules**” to compute answers.

Where is 101worker?

- **GitHub** https://github.com/101companies/101worker
- **Git clone** https://github.com/101companies/101worker.git
- **Doc** https://github.com/101companies/101worker/blob/master/README.md
- **Latest results online** http://data.101companies.org/

I/O behavior



One module's output may be the next module's input*!*

How to run 101worker?

- It's run for you in the cloud.
- Results are online: http://data.101companies.org/
- You could run the 101worker locally, **if you really wanted to**.
  - Follow the doc.
  - You should do this on an ubuntu machine.
- **You could run 101worker in a test environment locally.**
  - Follow the doc and see the following slides.
  - This works best on ubuntu & Mac OS X; it works on Windows, too.

Prepare 101worker for running it in a test environment locally

- Make sure you have Python3. You may need pip3.
- *git clone* https://github.com/101companies/101worker.git
- *cd 101worker* (change directory)
- *make init -B* (or create directories according to Makefile)
- On ubuntu: *sudo make install-debian-pkgs*
- Get pip packages: *sudo make install-pip-pkgs*
- Download latest 101worker data: *make download*

Let’s look at modules, primary resources, derived resources, and dumps.

- Assumptions:
  - The preparation steps were completed.
  - Filenames are relative to “101worker” directory.
- Sample modules:
  - simpleLoc: simple LOC metric per primary resource
  - locPerContribution: LOC sum per contribution in a dump
  - languageFrequency: Language frequencies as used on wiki
  - packageFrequency: Java package frequency as imported

Run modules in test environment locally

- Command line:
  - *bin/run_module simpleLOC*
- List of included contributions:
  - config/test_folders.txt
- The run_module script copies contributions to *101test*.

Module *simpleLOC*

https://github.com/101companies/101worker/tree/master/modules/simpleLOC

- **Summary**: For each *primary* resource, a resource with extension .loc.json is *derived*; it states the LOC metric ("lines of code") for the primary resource.

- **Example of consumed primary resource**:
  - ../101repo/contributions/py3k/Company.py

- **Example of produced derived resource**:
  - ../101web/data/resources/contributions/py3k/Company.py.loc.json

Module *locPerContribution*

https://github.com/101companies/101worker/tree/master/modules/locPerContribution

- **Summary**: The derived .loc.json resources are grouped by contribution and the resulting LOC sums per contribution are saved in a dump.

- **Example of consumed derived resource**:
  - ../101web/data/resources/contributions/py3k/Company.py.loc.json

- **Produced dump**:
  - ../101web/data/dumps/locPerContribution.json

Module *languageFrequency*

https://github.com/101companies/101worker/tree/master/modules/languageFrequency

- **Summary**: The 101wiki triples of the form “contribution uses language” are counted per language and resulting language frequencies are saved in a dump.

- **Consumed wiki dump**:
  - ../101web/data/dumps/wiki.json

- **Produced dump**:
  - ../101web/data/dumps/languageFrequency.json

Module *packageFrequency*

https://github.com/101companies/101worker/tree/master/modules/packageFrequency

- **Summary**: From the derived .extractor.json resources the imported packages are extracted and the frequencies of import per package are saved in a dump. This is only done for primary resources that a Java files, which is determined on the grounds of derived .lang.json resources.
- **Examples of consumed derived resources**:
  - ../101web/data/resources/contributions/javaComposition/src/main/java/org/softlang/company/model/Company.java.**lang.json**
  - ../101web/data/resources/contributions/javaComposition/src/main/java/org/softlang/company/model/Company.java.**extractor.json**
- **Produced dump**:
  - ../101web/data/dumps/packageFrequency.json

The module contract

- See the source code of the mentioned modules.
- See the documentation.

- Key methods of modules:
  - *run* method processes primary resources.
  - *test* method tests module on mocked resources/dumps.

Questions

- Please use Facebook group.
- Help each other.
- Assistants also try helping on Facebook.