Wanderlust
Cache Warmer with priorities

The Wanderer
Behaviour
- Gets list of URLs and dispatches them to the workers
- Returns Time for downloading and size for each URL

Worker Pool 🧩
- HTTP Worker
- HTTP Worker
- HTTP Worker

The Wanderer
– Reads new URLs from the DB, fetches them and saves the results into the DB.
– Marks fetched URLs as already processed
– Goroutine, r/w and watch to DB

Provisioner
Provisioner
– List of modules which provides the URLs
– Maintain Blacklist of URLs (e.g. no checkout)
– Configure Priorities

Prio ❶ — Google Analytics
- Top 100 URLs last 2 weeks
- Top 200 sold products last 2 weeks

Piwik
- Top 100 URLs last x weeks
- Top 200 sold products last x weeks

Prio ❷ — JSON REST Endpoints with/without OAuth config
- e.g. Top 200 Sold products in Magento
- e.g. CMS Pages
- all other non-important products/pages
- products/pages with the last cleared cache

JSON Format
```
{
    "items": [
        {
            "url": "http://www.a.com/catalog/product/view/123"
        },
        {
            "url": "http://www.a.com/about-us"
        }
    ]
}
```

Prio ❸ — sitemap.xml

Prio ❹ — Manually added URLs

Prio ⓪ — Crawl all the remaining URLs of a website
If configured

Picnic
Picnic
The web interface
– Watches DB for new URLs and already processed.
– Pushes results from DB to the user via Websockets
– Handles configurations

Actions
- Start the Wanderer
- Start the Brotzeit
- Purge URLs collected from Brotzeit by provisioner
- Purge all data

Views
- View performance metrics
- View all URLs from the provisioners
- View Amount of Running Background Workers

Configurations
- Provisioners and their access data
- Wanderer: Concurrency level
- Brotzeit: Concurrency level

The Rucksack
The Rucksack
KV? Database for storage. No 3rd party. Compiled into.
- Write access to HDD only for saving

General Stuff
- Future Features — Distributed Wanderers
- The only write access to the HDD is for saving the DB
- Wanderlust can run anywhere
- All CSS, JS, HTML files are compiled into

Brotzeit
Brotzeit
– Fetches all the URLs from the Provisioner and stores them in the DB
– Automatically fills the DB as soon as access data is provided to a provisioner or Enable: yes/no
– Goroutine