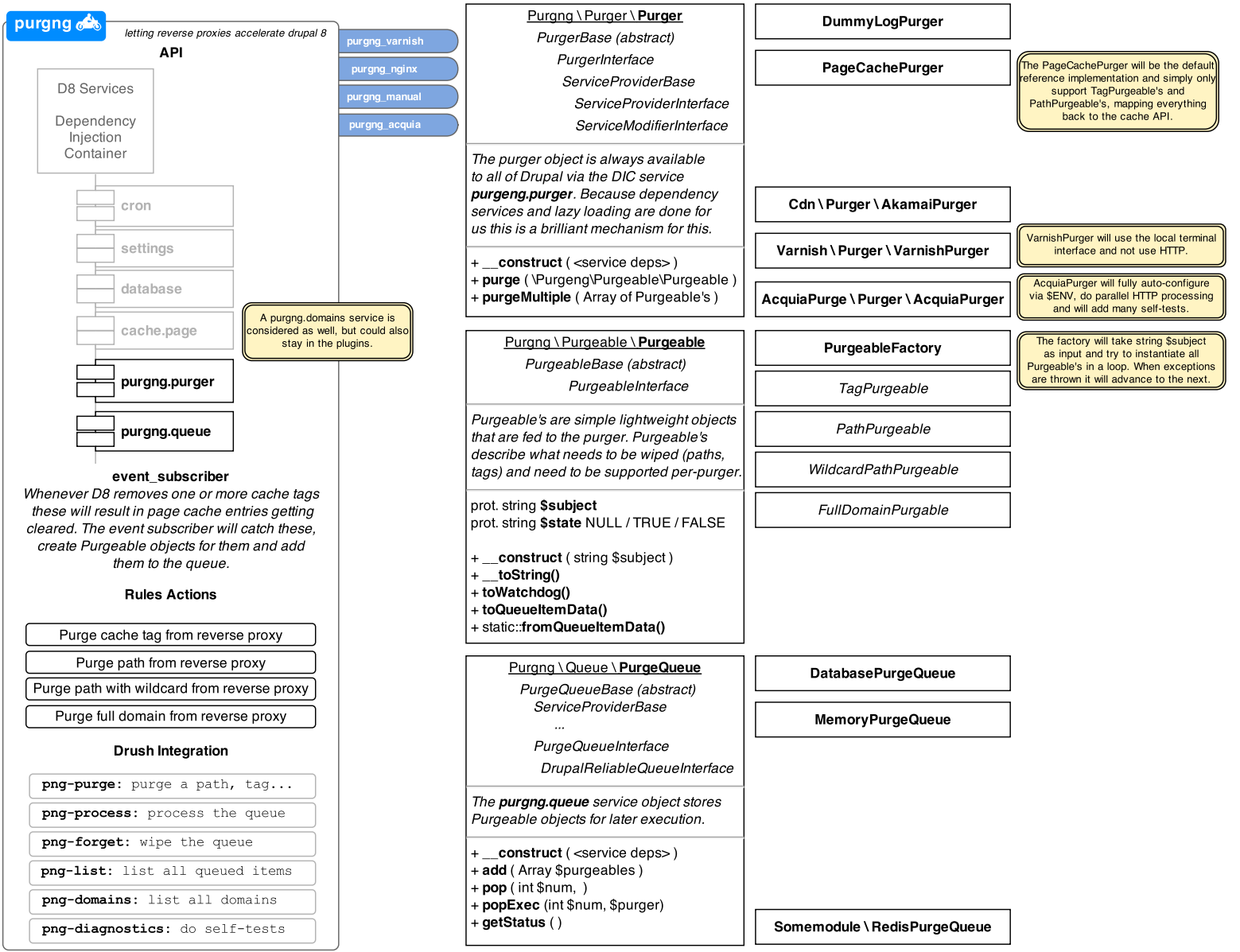
purgng
letting reverse proxies accelerate drupal 8
API
D8 Services
Dependency Injection Container
cron
settings
database
cache.page
purgng.purger
purgng.queue
purgng_varnish
purgng_nginx
purgng_manual
purgng_acquia
A purgng.domains service is considered as well, but could also stay in the plugins.
event_subscriber
Whenever D8 removes one or more cache tags these will result in page cache entries getting cleared. The event subscriber will catch these, create Purgeable objects for them and add them to the queue.
Rules Actions
Purge cache tag from reverse proxy
Purge path from reverse proxy
Purge path with wildcard from reverse proxy
Purge full domain from reverse proxy
Drush Integration
png-purge: purge a path, tag...
png-process: process the queue
png-forget: wipe the queue
png-list: list all queued items
png-domains: list all domains
png-diagnostics: do self-tests
Purgng \ Purger \ Purger
PurgerBase (abstract)
PurgerInterface
ServiceProviderBase
ServiceProviderInterface
ServiceModifierInterface
The purger object is always available to all of Drupal via the DIC service purgeng.purger. Because dependency services and lazy loading are done for us this is a brilliant mechanism for this.
+ __construct (<service deps>)
+ purge (\Purgeng\Purgeable\Purgeable)
+ purgeMultiple (Array of Purgeable's)
Purgng \ Purgeable \ Purgeable
PurgeableBase (abstract)
PurgeableInterface
Purgeable's are simple lightweight objects that are fed to the purger. Purgeable's describe what needs to be wiped (paths, tags) and need to be supported per-purger.
prot. string $subject
prot. string $state NULL / TRUE / FALSE
+ __construct (string $subject)
+ __toString()
+ toWatchdog()
+ toQueueItemData()
+ static::fromQueueItemData()
Purgng \ Queue \ PurgeQueue
PurgeQueueBase (abstract)
ServiceProviderBase
...
PurgeQueueInterface
DrupalReliableQueueInterface
The purgng.queue service object stores Purgeable objects for later execution.
+ __construct (<service deps>)
+ add (Array $purgeables)
+ pop (int $num,)
+ popExec (int $num, $purger)
+ getStatus ( )
DummyLogPurger
PageCachePurger
The PageCachePurger will be the default reference implementation and simply only support TagPurgeable's and PathPurgeable's, mapping everything back to the cache API.
Cdn \ Purger \ AkamaiPurger
Varnish \ Purger \ VarnishPurger
VarnishPurger will use the local terminal interface and not use HTTP.
AcquiaPurge \ Purger \ AcquiaPurger
AcquiaPurger will fully auto-configure via $ENV, do parallel HTTP processing and will add many self-tests.
PurgeableFactory
The factory will take string $subject as input and try to instantiate all Purgeable's in a loop. When exceptions are thrown it will advance to the next.
TagPurgeable
PathPurgeable
WildcardPathPurgeable
FullDomainPurgable
DatabasePurgeQueue
MemoryPurgeQueue
Somemodule \ RedisPurgeQueue