Sample Title

Switch to Magento 2

Staff, techniques, tricks and some Python

DevelopersParadise
2016 / Opatija / Croatia

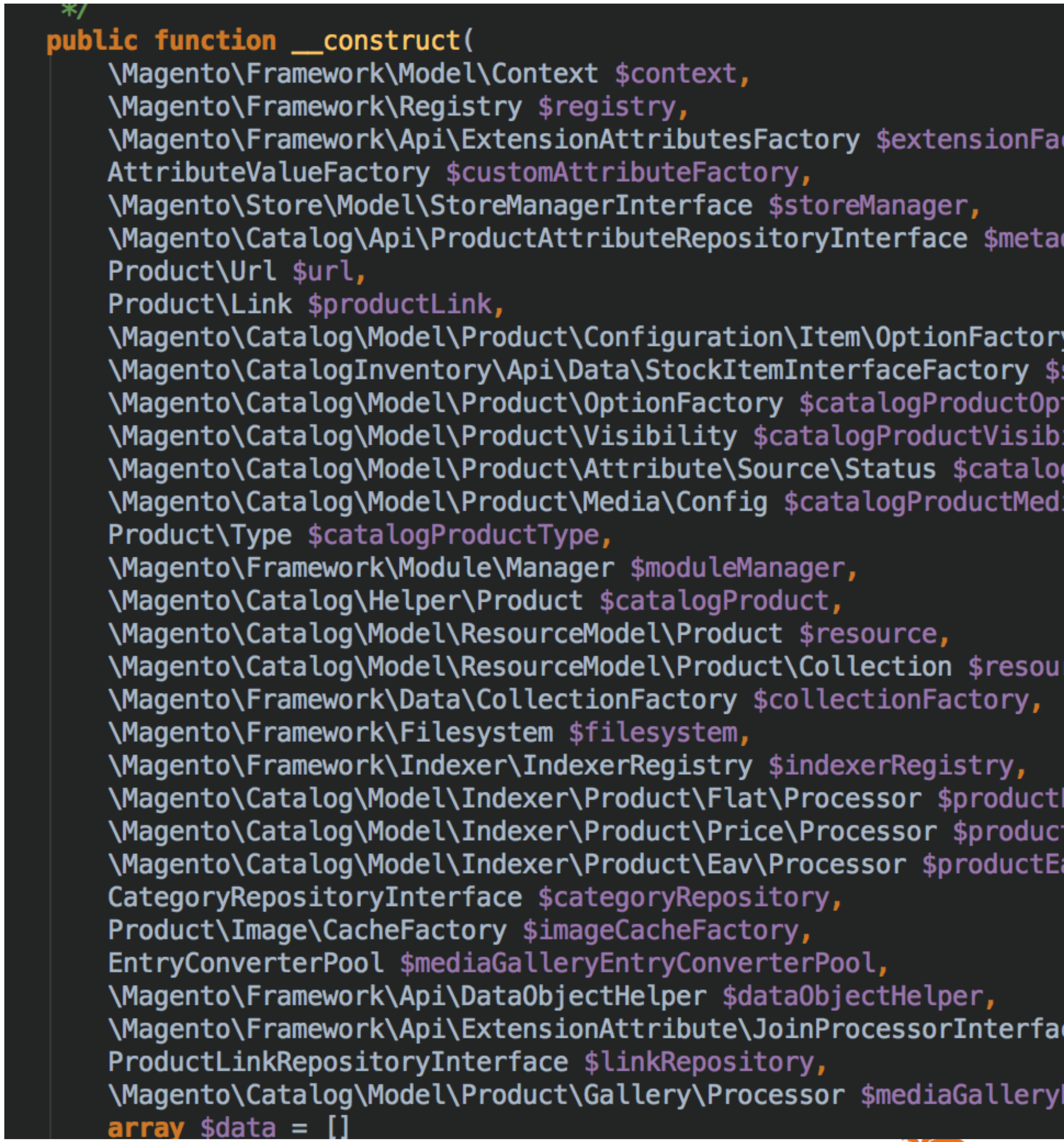
Peter Samoilov

CTO - aheadWorks

About me and aheadWorks

- Being with Magento from the very start
- I hate coding
- I hate PHP
- \Especially\Namespaces
- We've already done a lot of mistakes with M2
- But we still plan to be #1 on extension market

Remember your first Magento 2 experience



```
    */
    public function __construct(
        \Magento\Framework\Model\Context $context,
        \Magento\Framework\Registry $registry,
        \Magento\Framework\Api\ExtensionAttributesFactory $extensionFa
        AttributeValueFactory $customAttributeFactory,
        \Magento\Store\Model\StoreManagerInterface $storeManager,
        \Magento\Catalog\Api\ProductAttributeRepositoryInterface $metad
        Product\Url $url,
        Product\Link $productLink,
        \Magento\Catalog\Model\Product\Configuration\Item\OptionFactory
        \Magento\CatalogInventory\Api\Data\StockItemInterfaceFactory $s
        \Magento\Catalog\Model\Product\OptionFactory $catalogProductOpt
        \Magento\Catalog\Model\Product\Visibility $catalogProductVisibi
        \Magento\Catalog\Model\Product\Attribute\Source\Status $catalog
        \Magento\Catalog\Model\Product\Media\Config $catalogProductMedi
        Product\Type $catalogProductType,
        \Magento\Framework\Module\Manager $moduleManager,
        \Magento\Catalog\Helper\Product $catalogProduct,
        \Magento\Catalog\Model\ResourceModel\Product $resource,
        \Magento\Catalog\Model\ResourceModel\Product\Collection $resou
        \Magento\Framework\Data\CollectionFactory $collectionFactory,
        \Magento\Framework\Filesystem $filesystem,
        \Magento\Framework\Indexer\IndexerRegistry $indexerRegistry,
        \Magento\Catalog\Model\Indexer\Product\Flat\Processor $productF
        \Magento\Catalog\Model\Indexer\Product\Price\Processor $product
        \Magento\Catalog\Model\Indexer\Product\Eav\Processor $productEa
        CategoryRepositoryInterface $categoryRepository,
        Product\Image\CacheFactory $imageCacheFactory,
        EntryConverterPool $mediaGalleryEntryConverterPool,
        \Magento\Framework\Api\DataObjectHelper $dataObjectHelper,
        \Magento\Framework\Api\ExtensionAttribute\JoinProcessorInterfa
        ProductLinkRepositoryInterface $linkRepository,
        \Magento\Catalog\Model\Product\Gallery\Processor $mediaGallery
        array $data = []
```

1. That’s not that old good PHP from Magento 1.x
2. Your team is a total noobs from now
3. There is no «Magento 2 developers» on outsourcing market
4. Other extension providers breathe down your neck

Hurry!

It’s a good time to make a bunch of mistakes!

Mistakes we made(or had to make). Some of.

1. Learn «on the job»
2. Apply your Magento 1 experience
3. Or make it like Magento 2 is done
4. Write tests after your product is finished
5. Or first write tests, then code
6. Stress your devs with strict deadlines

You don’t need your products ASAP.
You need to gain expertise ASAP!

The guys you have

1. You need masterminds only
2. Don’t deal with retards

The guys you want to have

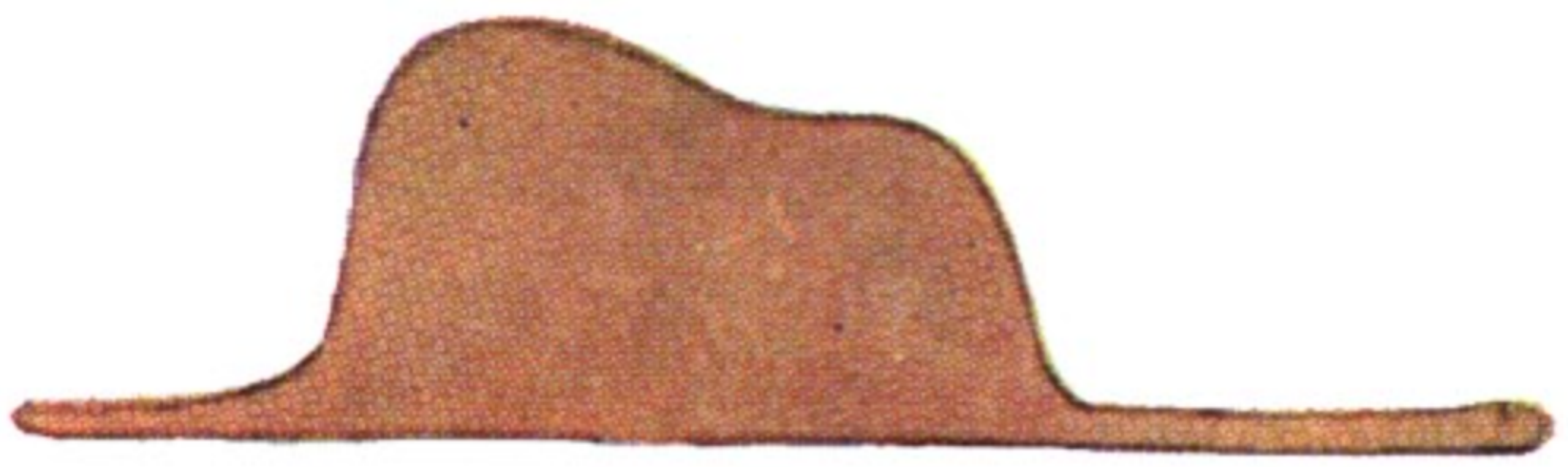
1. Good PHP dev studies M2 faster, than perfect M1 dev
2. Symfony2-skilled guys are great for this job
3. MVVM frameworks experience

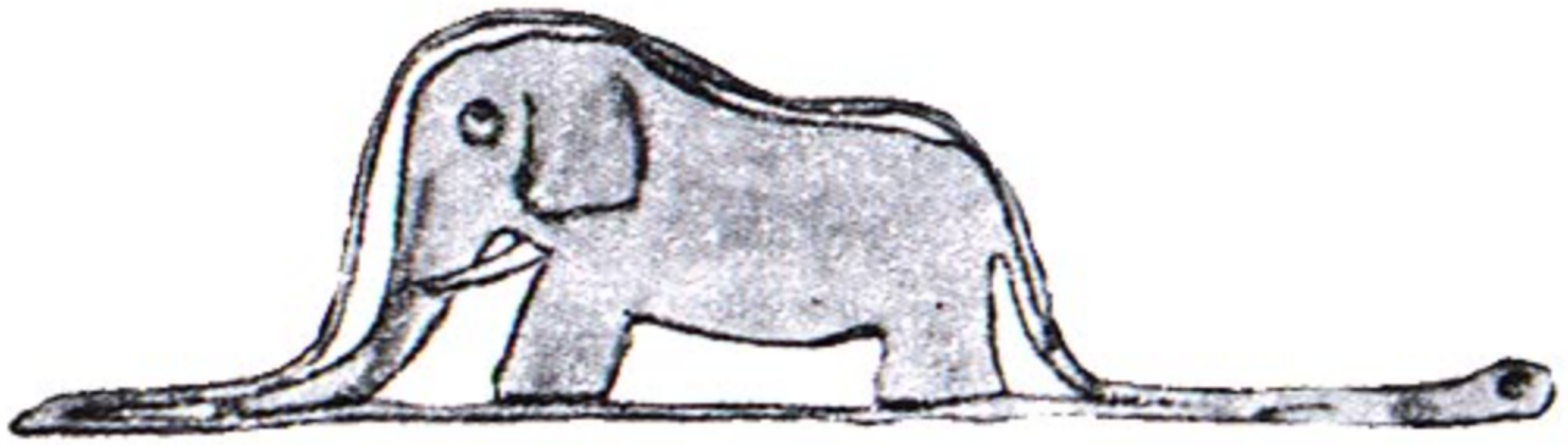
Some tricks

1. Study Magento 2 basic design patterns
    1. Dependency Injection/IoC
    2. Loose coupling
    3. Service contracts
    4. Interceptors
2. Perform couple of small/middle side projects with M2
3. Start with Magento 2 Test Framework, not M2 itself
4. Always use bleeding edge M2 revision
5. Watch inside module-customer
6. Automate all what you see

What’s on your laptop?

How snakes can help elephants?





Some great things about Python

1. It’s everywhere
2. It’s simple
3. Ready recipes for almost everything

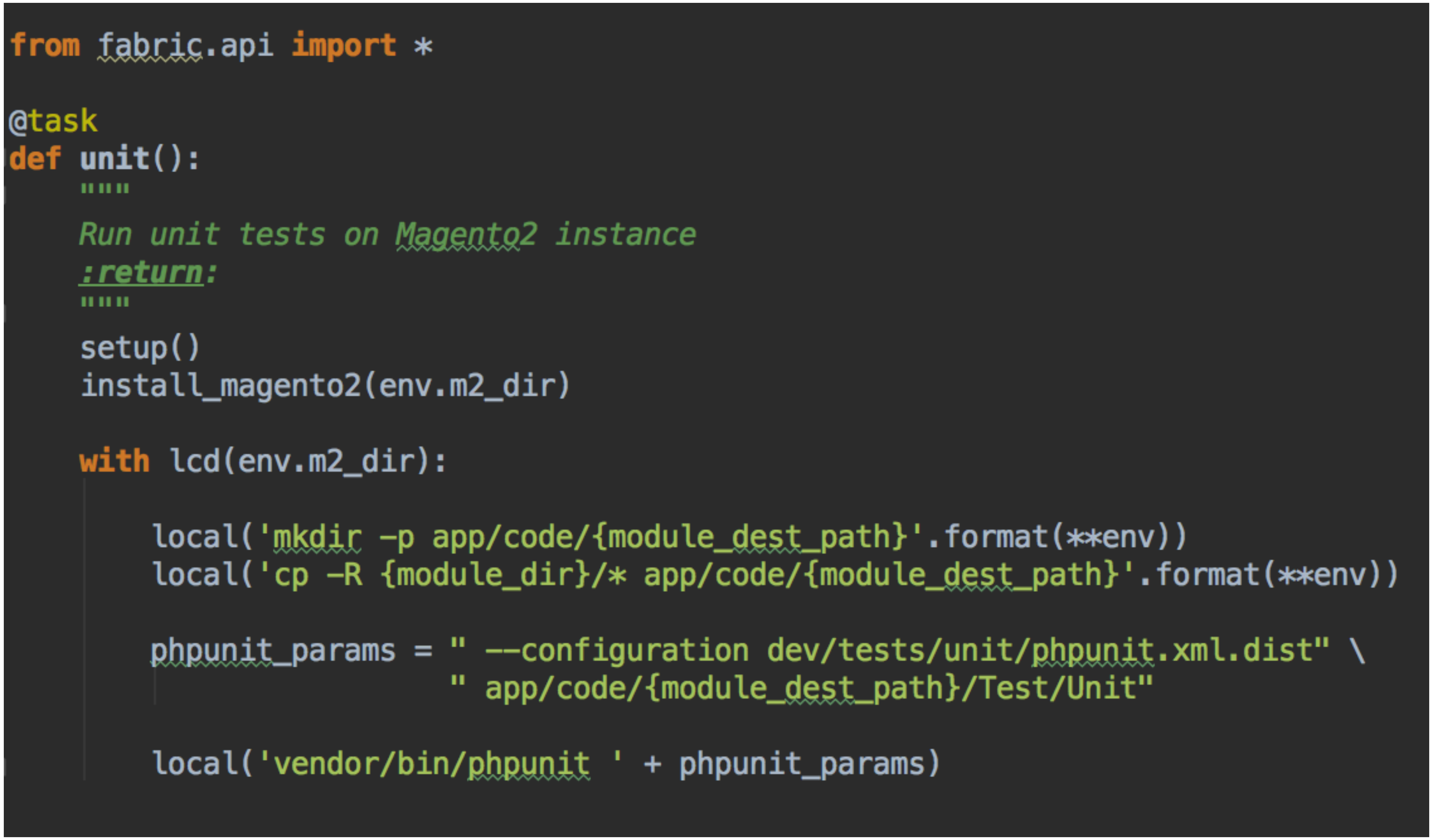
```
import __hello__
```

Fabric

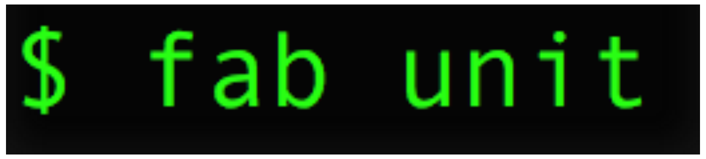
1. API for streaming the use of SSH
2. CLI for calling Python methods
3. Simple even for starter

```
$ pip install fabric
```

Fabric



```
from fabric.api import *

@task
def unit():
    """
    Run unit tests on Magento2 instance
    :return:
    """
    setup()
    install_magento2(env.m2_dir)

    with lcd(env.m2_dir):

        local('mkdir -p app/code/{module_dest_path}'.format(**env))
        local('cp -R {module_dir}/* app/code/{module_dest_path}'.format(**env))

        phpunit_params = " --configuration dev/tests/unit/phpunit.xml.dist" \
                         " app/code/{module_dest_path}/Test/Unit"

        local('vendor/bin/phpunit ' + phpunit_params)
```



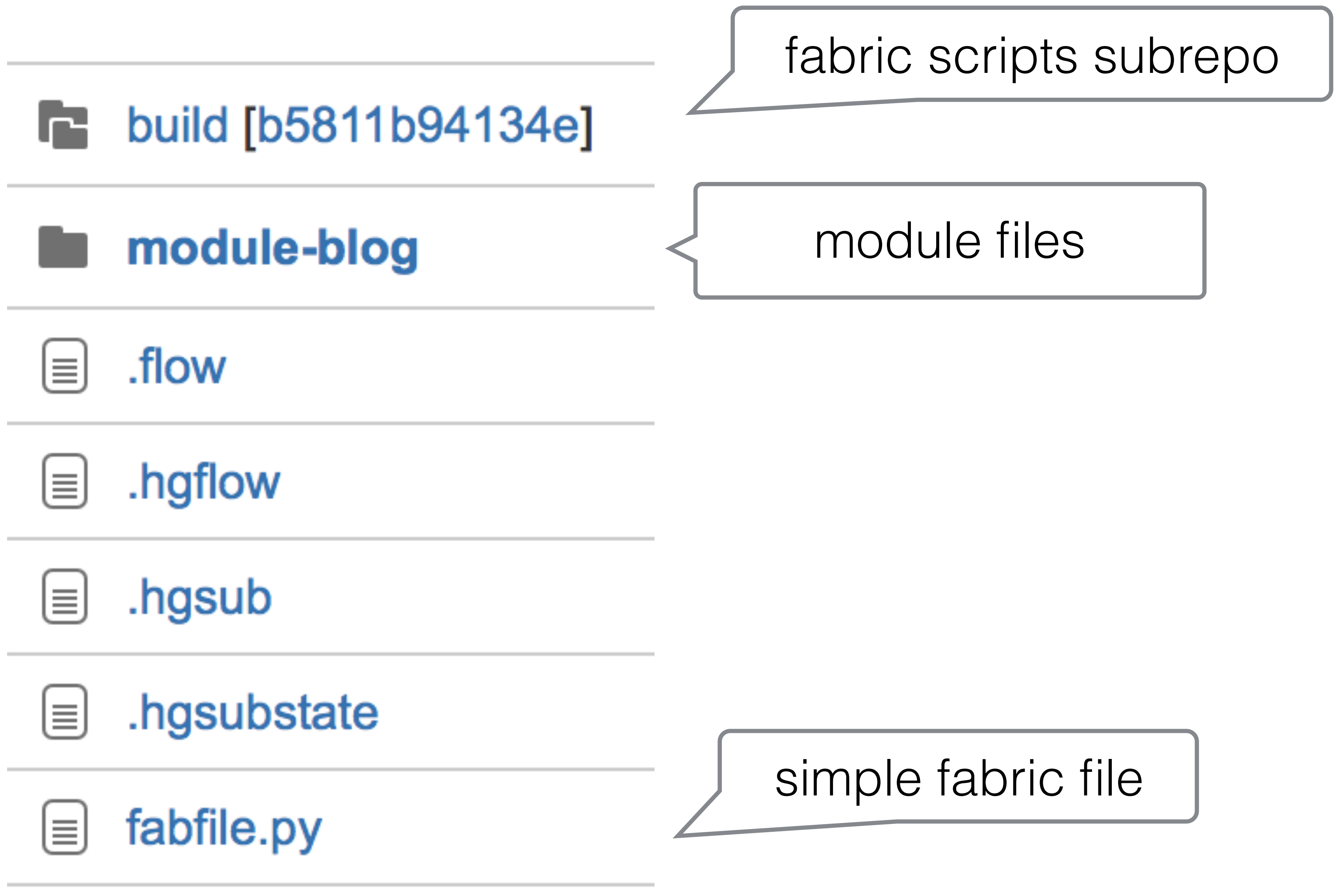
```
$ fab unit
```

DevelopersParadise
2016 / Opatija / Croatia

Fabric - more fun!

1. Put your fabric scripts as subrepo to all your extensions
2. Use same commands for all your extensions to:
   1. Run tests
   2. Build packages
   3. Etc, etc, etc

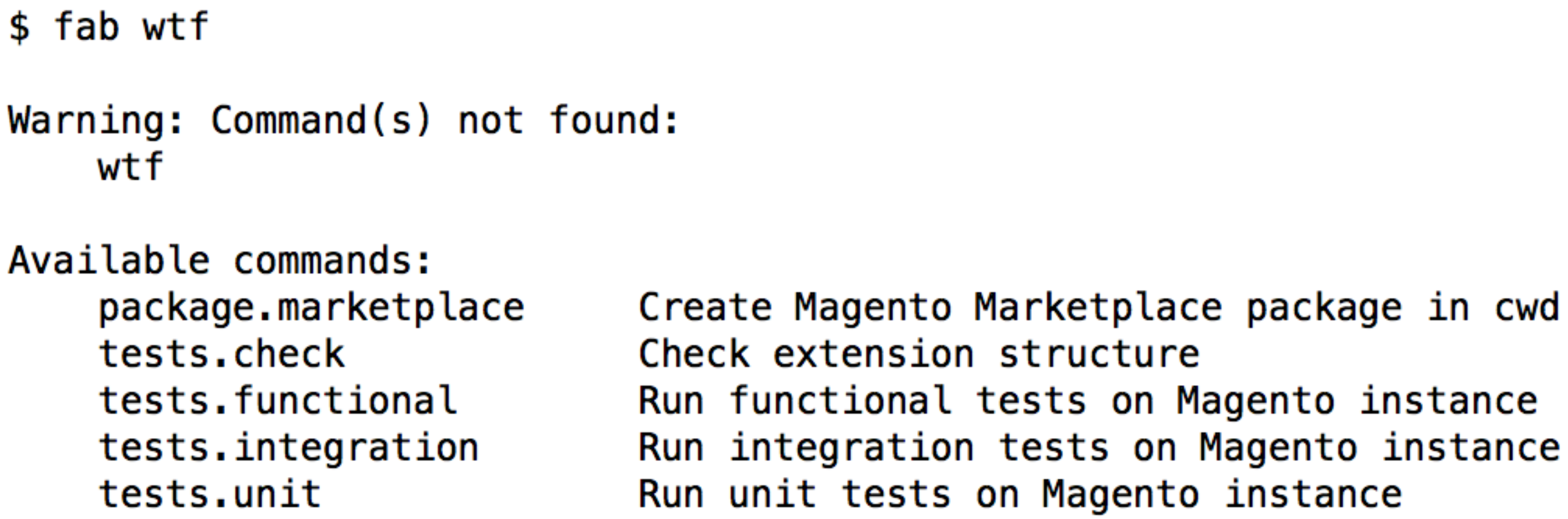
Fabric - more fun!



build [b5811b94134e]
module-blog
.flow
.hgflow
.hgsub
.hgsubstate
fabfile.py

```
from build.magento2 import *
```

Fabric - more fun!



```
$ fab wtf

Warning: Command(s) not found:
    wtf

Available commands:
    package.marketplace     Create Magento Marketplace package in cwd
    tests.check             Check extension structure
    tests.functional        Run functional tests on Magento instance
    tests.integration       Run integration tests on Magento instance
    tests.unit              Run unit tests on Magento instance
```

Nice tool for any kind of automations for your projects

Share it between all your projects

Sphinx and RST

http://www.sphinx-doc.org/

1. The only good .rst processor
2. Much better than .md(can handle docs references at least)
3. Can highlight php and other syntaxes
4. Has bunch of plugins almost for all (even UML diagrams!)
5. Best for docs stored with project files

```
$ pip install sphinx
```

Docs strategies

1. Store your docs source inside version control
2. Schedule a time for docs updates
3. Modify it with the code - it's a part of a project
4. Mix it with fabric(or similar tool) to compile it in CI or any other moment
5. Expose compiled docs somewhere

“ That was a cool slide provided within template.

I decided to leave it here. ”

Thank You!

DevelopersParadise

2016 / Opatija / Croatia