Blog

5分で分かるDockerのキホン

2014/05/25

2014 devops docker packer vagrant 寿司

全国100万人のImmutable Infrastructure職人のみなさんこんにちは。 もう誰も彼もがDockerなので、あんまりブログに書こうという気にもならなかったのですが、知り合いからリクエストを貰ったので、5分くらいで分かるようにかいつまんで概略を説明します。

Dockerとは

- 詳しくは本家サイト見ればだいたい分かる。
- 仮想化技術
- コンテナ単位でパッケージング
  - VirtualBoxとかと違って高速、オーバーヘッドが少ない。chrootに近い。LXCには依存しなくなっている
  - コンテナごとにIDが振られる
  - コンテナは差分保存なのでロールバックも簡単
- 一回作ればどこでも動く。Javaっぽい
- Dockerfileでコンテナを作成する
  - Dockerfileの1行ごとにコンテナIDがフラれる
- 動作環境
  - Linux Kernel 3.8以降 64bit OS
  - Macの場合はVirtualBoxの中で動かす形になる→ boot2docker
  - Windowsの場合もVirtualBoxを使って仮想マシンの中にDocker環境を作ることになる。
- 何がうれしいの？
  - 高速に起動する。したがってCI用に使ったり、Chefのクックブックの実験環境に使ったり、開発環境に使ったり色々便利。もう本番環境で使っている例も多数
  - コンテナ内に閉じ込めることでポータビリティがあがる。アプリケーションのデプロイ戦略が楽な方に大きく変わりうる
  - 差分で管理されるので配布と再利用が簡単
  - いろんな環境で動く
  - Dockerfileを使ってコードでインフラを管理できる

Ubuntu 12.04 LTS 64bitで動かす

まずKernelを3.8に変更する

```
sudo apt-get update
sudo apt-get install linux-image-generic-lts-raring linux-headers-generic-lts-raring
sudo reboot
```

aptレポジトリを追加してインストール

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 36A1D7869245C8950F966E92D8576A8BA88D21E9
sudo sh -c "echo deb https://get.docker.io/ubuntu docker main /etc/apt/sources.list.d/docker.list"
sudo apt-get update
sudo apt-get install lxc-docker
```

とりあえずDockerを使ってUbuntuのコンテナを起動する

```
sudo docker run -i -t ubuntu /bin/bash
```

初回なのでたくさんイメージをダウンロードするのでちょっと時間がかかるはず。終わると以下のようにログインできている

```
Unable to find image 'ubuntu' locally
Pulling repository ubuntu
316b678ddf48: Download complete
a7cf8ae4e998: Download complete
3db9c44f4520: Download complete
(略)
cb12405ee8fa: Download complete
4d26dd3ebc1c: Download complete
d4010efcfd86: Download complete
root@6485cdbce682:/#
```

この状態だと、インタラクティブシェルで、exitするとプロセスが終了する。

どのコンテナが起動しているかを確認する

```
sudo docker ps
```

として確認すればOK。(ps -a だと停止済のコンテナも含む)

コンテナ自体を削除する場合

以下のようにする（末尾のパラメータはコンテナID）

```
sudo docker rm 04b26990125f
```

複数のコンテナをまとめて停止したり削除する

以下のように引数にコマンド実行の結果を渡せばOK。

```
sudo docker stop `sudo docker ps -aq`
sudo docker rm `sudo docker ps -aq`
```

独自のイメージを作る

適当に変更を加えたあとにログアウトし、

```
sudo docker commit コンテナID 適当な名前
```

のようにする。作ったものを確認するには

```
sudo docker images 名前
```

として一覧に出ることを確認すればOK

たとえば

```
sudo docker commit eb0dd1195df2 ryuzee/apache2
```

これができれば

```
sudo docker run -i -t ryuzee/apache2 /bin/bash
```

にように今度はこれを使うことができる。なお、イメージの名称は、ユーザー名/中身 という形式にするのが推奨

イメージの一覧を取得する

```
sudo docker images
```

とすることでイメージの一覧が取得できる。

```
REPOSITORY      TAG         IMAGE ID        CREATED         VIRTUAL SIZE
ubuntu          13.10       5e019ab7bf6d    4 weeks ago     180 MB
ubuntu          saucy       5e019ab7bf6d    4 weeks ago     180 MB
ubuntu          12.04       74fe38d11401    4 weeks ago     209.6 MB
(略)
ubuntu          10.04       3db9c44f4520    4 weeks ago     183 MB
ubuntu          lucid       3db9c44f4520    4 weeks ago     183 MB
```

Dockerfileでイメージを作る

```
FROM ubuntu
RUN sudo apt-get update -y
RUN sudo apt-get install apache2 -y
RUN sudo apt-get install memcached -y
RUN sudo apt-get install php5 -y

## 環境変数の設定
ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_PID_FILE /var/run/apache2.pid
ENV APACHE_RUN_DIR /var/run/apache2
ENV APACHE_LOG_DIR /var/log/apache2
ENV APACHE_LOCK_DIR /var/lock/apache2

## 公開するポートの指定
EXPOSE 80

## コンテナ起動時に実行するコマンドを指定
CMD ["/usr/sbin/apache2", "-D", "FOREGROUND"]
```

のように、ベースとするイメージを指定したあと、実行するコマンドを書いていく。これができたら

```
sudo docker build .
```

を実行するとビルドされる。ただしこれで作られたイメージには名前がついていないので、

```
sudo docker build -t 名前 .
```

のようにすると良い。もしくはあとから名前をつける場合は、

```
sudo docker tag bc9e46f7accc ryuzee/apache2
```

のようにする

Dockerfile内で使えるコマンド

```
FROM <image> 元となるイメージを指定する。:の後ろでタグ指定可能
MAINTAINER <name> メンテナの名前
RUN <command> ビルド中に実行したいコマンドを指定。パッケージのインストールとか設定とか
CMD <command> 起動後のコンテナで実行したいコマンドを指定する。1つしか書けない
EXPOSE <port> [</port><port> ...] 外部に公開するポートを指定
ENV <key> <value> 環境変数の設定
ADD <src> <dest> ファイルを配置。絶対パスで記述する
```

Dockerfile作成上の注意

- ENTRYPOINTやCMDは1つのDockerfileの中で1個しか指定できない

- したがって複数プロセスを起動したい時はsupervisordを使って行う

イメージの削除

以下のように引数にイメージIDを指定する。ただし依存するコンテナが存在する場合は削除できない

```
sudo docker rmi f6eb3276aab2
```

ポートの設定

Dockerfileの中でEXPOSEを使って指定するか、起動時に設定する。以下の例では、コンテナ側の80番にポートフォワードするために母艦側で8888を指定している

```
sudo docker run -d -p 8888:80 ryuzee/apache2
```

起動サービスの設定

Dockerfileの中でCMDを使って指定するか、起動時に設定する

```
docker run -d -p 8080:80 ryuzee/apache2 /usr/sbin/apache2 -DFOREGROUND
```

ローカル側のディスクを共有

あらかじめDockerfileの中で公開するVOLUMEの設定をしておいた上で、起動時に -v を使ってローカル側:コンテナ側の形で指定する。なお、絶対パスで指定する必要がある点には注意。

ログ系とかは外だしにしとくと便利かも。

```
sudo docker run -p 9999:80 -p 2222:22 -t -v /home/ryuzee/docker/basic/logs:/var/log/apache2 -d ryuzee/basic
```

イメージの共有

Docker Registryを使って作成したイメージを共有することができる。例えば、パブリックレジストリを使うと世の中に公開したり、他の人が公開しているものを使ったりできる

公式レポジトリとコマンドラインをひもづけるには

```
sudo docker login
```

としてログインすればOK

作ったイメージを公開するには、

```
sudo docker push ryuzee/basic
```

としてプッシュしてあげれば良い。

以下のようなログが表示される。

```
The push refers to a repository [ryuzee/basic] (len: 1)
Sending image list
Pushing repository ryuzee/basic (1 tags)
511136ea3c5a: Image already pushed, skipping
Image f10ebce2c0e1 already pushed, skipping
Image 82cdea7ab5b5 already pushed, skipping
Image 5dbd9cb5a02f already pushed, skipping
```

```
Image 74fe38d11401 already pushed, skipping
869d76ced5c7: Image successfully pushed
(略)
424924b964f9: Image successfully pushed
4f0307fd0f8f: Image successfully pushed
d77d7fd91796: Image successfully pushed
532d67a9eec2: Image successfully pushed
5804bf0385b9: Image successfully pushed
Pushing tag for rev [5804bf0385b9] on {https://registry-1.docker.io/v1/repositories/ryuzee/basic/tags/latest}
```

自分用のDocker registryをたてる

index.docker.io だけでなく、自分でDocker Registryを作ることができる。またこのとき、Amazon S3にファイルが保存できるので、AWS内に仮想マシンをたててDocker使う場合はネットワーク的なメリットもある。

手順

まずS3に適当にバケットを作成しておく。今回は東京リージョンにryuzee-dockerimageという名前で作成。なお、docker pullして持ってくるregistryは古いので、できればGitHubから最新を持ってきたほうが困らない感じ。

```
git clone https://github.com/dotcloud/docker-registry
cd docker-registry
sudo docker build -t ryuzee/registry .
```

コンテナを起動する。-eで環境変数を指定しており、アプリの中で必要なAWSのアクセスキーやシークレットキーを指定している。もちろんコンテナの中にconfig.ymlを置いたり、起動時にVOLUMEを使って設定ファイルを外部から共有してもOK。

```
sudo docker run \
      -e SETTINGS_FLAVOR=s3 \
      -e AWS_BUCKET=ryuzee-dockerimage \
      -e STORAGE_PATH=/images \
      -e AWS_KEY=AKIXXXXXXXXXXXXXXXXX \
      -e AWS_SECRET=abcnjsifhAHFkfkjidjdjdjdiki \
      -e SEARCH_BACKEND=sqlalchemy \
      -p 5000:5000 \
      ryuzee/registry
```

ここまでできたら、まず以下のようにタグをつけて

```
sudo docker tag bb34bb42eb3a localhost:5000/ryuzee/basic
```

その後にレポジトリにプッシュする

```
sudo docker push locahost:5000/ryuzee/basic
```

Pullしたいときの考えかたも基本的に一緒

```
sudo docker pull localhost:5000/ryuzee/basic
```

dockerコマンドでsudo毎回叩くとかありえない人向け

dockerグループに自分を追加してあげることでsudo付けなくてもOKになる。なお作業完了後、シェルの再起動が必要なので、ログアウト/ログインする

```
sudo groupadd docker
sudo gpasswd -a ryuzee docker
sudo service docker restart
```

Ubuntu上のVagrantから簡単に操作する

Vagrant 1.6系からDockerに正式対応したので、Ubuntu上だとかなり楽に使える。 Vagrantfileは以下のようになる。

```
Vagrant.configure("2") do |config|
 config.vm.provider "docker" do |d|
  d.image = "ryuzee/basic"
 end
end
```

これを用意して、いつもどおり

```
vagrant up --provider=docker
```

とすればOK

vagrant sshでログインしてごにょごにょしたい場合は以下のようにhas_sshを有効にしつつ、sshでの接続ユーザー名やキーペア(またはパスワード)とポート番号を指定する(ポート番号を指定しない場合、なぜかBad Portというエラーになるので注意)

```
Vagrant.configure("2") do |config|
 config.vm.provider "docker" do |d|
  d.image = "ryuzee/basic"
  d.remains_running = false
  d.has_ssh = true
 end
 config.ssh.username = "root"
 config.ssh.password = "root"
 config.ssh.port = "22"
end
```

さらにコンテナ起動時のCMDを指定したり、ポートマッピングを設定することも可能。

```
## このあたりが追加になる
d.cmd = ["/usr/sbin/httpd", "-DFOREGROUND"]↵
d.ports = ["8888:80"]
```

過去の資産の再利用

Chefとの組み合わせ方

packerを使ってイメージの作成を行えば、過去の資産の再利用が可能。Packerは0.6以上のバージョンにすること。 以下の例では、BerkshelfおよびChef-Soloプロビジョナーとの組み合わせで、Dockerのイメージを作成している。

```
{
 "builders": [{
  "type": "docker",
  "image": "centos",
  "export_path": "image.tar"
 }],
 "provisioners":[
 {
  "type": "chef-solo",
  "cookbook_paths": ["./vendor-cookbooks/"],
  "run_list": ["timezone", "apache2-simple", "memcached"],
  "json": {"memcached":{"maxcon":"512","cachesize":"512"}},
  "prevent_sudo": true,
  "skip_install": false
```

```
  }],
  "post-processors": [{
    "type": "docker-import",
    "repository": "ryuzee/packer-sample",
    "tag": "0.1"
  }]
}
```

ソースはこちら。 一点気をつける点としては、Dockerfileでサポートされ ているEXPOSEやENDPOINTやCMDなどを現時点ではサポートしていないため、自分でdocker runの際に引数で渡す必要がある。

あとは頼んだ…

2014/05/25

2014 devops docker packer vagrant 寿司
Grafanaを使ってGraphiteのデータを表示するダッシュボードを作る
Sensu on AWSな話

Effective DevOps ―4本柱による持続可能な組織文化の育て方

*著者／訳者：*Jennifer Davis、Ryn Daniels / 吉羽 龍太郎、長尾高弘

*出版社：*オライリージャパン(2018-3-24)

定価：￥ 3,888

主にDevOpsの文化的な事柄に着目し、異なるゴールを持つチームが親和性を高め、矛盾する目標のバランスを取りながら最大限の力を発揮する方法を解説します

変革の軌跡【世界で戦える会社に変わる"アジャイル・DevOps"導入の原則】

*著者／訳者：*Gary Gruver、Tommy Mouser / 吉羽 龍太郎、原田騎郎

*出版社：*技術評論社(2017-1-25)

定価：￥ 2,138

米ヒューレット・パッカードのLaserJet部門でいかにしてAgileやDevOpsの原則を適用しマーケットの変化に対応できるようになったかの事例を解説。巻末には日本の大手企業の事例も収録

ジョイ・インク 役職も部署もない全員主役のマネジメント

*著者／訳者：*リチャード・シェリダン / 原田騎郎, 安井力, 吉羽龍太郎, 永瀬美穂, 川口恭伸

*出版社：*翔泳社(2016-12-20)

定価：￥ 1,944

米国で何度も働きやすい職場として表彰を受けているメンローの創業者かつCEOであるリチャード・シェリダン氏が、職場に喜びをもたらす知恵や経営手法、より良い製品の作り方などを惜しみなく紹介しています

Amazon Web Services企業導入ガイドブック

*著者／訳者：*荒木靖宏, 大谷晋平, 小林正人, 酒徳知明, 高田智己, 瀧澤与一, 山本教仁, 吉羽龍太郎

*出版社：*マイナビ出版(2016-06-10)

定価：￥ 2,797

AWSを企業に導入する際に知っておくべきAWSサービスの全貌から、セキュリティ概要、システム設計、導入プロセス、運用までをまとめた書籍。操作の仕方ではなくどのように導入していくかの考え方や設計方法に焦点をあてています

アジャイルコーチの道具箱 – 見える化の実例集

*著者／訳者：*Jimmy Janlén / 原田騎郎, 吉羽龍太郎, 川口恭伸, 高江洲睦, 佐藤竜也

*出版社：*Leanpub(2016-04-12)

定価：$14.99

この本は、チームの協調とコミュニケーションを改善したり、行動を変えるための見える化の実例を集めたものです。96個(+2)の見える化の方法をそれぞれ1ページでイラストとともに解説しています。アジャイル開発かどうかに関係なくすぐに使えるカタログ集です

カンバン仕事術 ―チームではじめる見える化と改善

*著者／訳者：*原田騎郎 安井力 吉羽龍太郎 角征典 高木正弘

*出版社：*オライリージャパン(2016-03-25)

定価：￥ 2,138

チームの仕事や課題を見える化する手法「カンバン」について、その導入から実践までを図とともにわかりやすく解説した書籍です。カンバンの原則や流れの管理などの入門的な事柄から、サービスクラス、メトリクスの使用、プロセスの改善など、一歩進んだ応用的な話題までを網羅的に解説します。カンバンを一から学びたい、組織で使ってみたいと考える方に最適な実践的な入門書です。

サーバ/インフラエンジニア養成読本 DevOps編 [Infrastructure as Code を実践するノウハウが満載!]

*著者：*吉羽龍太郎 新原雅司 前田章 馬場俊彰

*出版社：*技術評論社(2016-02-26)

定価：￥ 2,138

DevOpsの基本と主にツール面からの解説。Ansibleによるサーバ管理、CircleCIでの継続的インテグレーションフロー、Dockerの話まで幅広いトピックを扱っています

SCRUM BOOT CAMP THE BOOK

*著者／訳者：*西村直人 永瀬美穂 吉羽龍太郎

*出版社：*翔泳社(2013-02-13)

定価：￥ 2,520

スクラム初心者に向けて基本的な考え方の解説から始まり、プロジェクトでの実際の進め方やよく起こる問題への対応法まで幅広く解説。マンガと文章のセットでスクラムを短期間で理解できます。スクラムの概要を正しく理解したい人、もう一度おさらいしたい人にオススメ。

CakePHPで学ぶ継続的インテグレーション

*著者／訳者：*渡辺 一宏 吉羽 龍太郎 岸田 健一郎 穴澤 康裕

*出版社：*インプレス(2014-09-19)

定価：￥ 4,320

Webアプリケーション開発における継続的インテグレーションについて、CakePHPのサンプルをベースにして、その概要から使用ツール解説、導入方法、メンテナンスまでを解説

Chef実践入門 ~コードによるインフラ構築の自動化 (WEB+DB PRESS plus)

*著者／訳者：*吉羽 龍太郎 安藤 祐介 伊藤 直也 菅井 祐太朗 並河 祐貴

*出版社：*技術評論社(2014-05-22)

定価：￥ 2,992

スタンドアロンでのChef Soloの利用から始めて、クックブックの書き方や注意すべきポイント、さらにはクックブックのテスト、継続的インテグレーションなど幅広いトピックを解説

Software in 30 Days スクラムによるアジャイルな組織変革“成功"ガイド

*著者／訳者：*Ken Schwaber、Jeff Sutherland著、角征典、吉羽龍太郎、原田騎郎、川口恭伸訳

*出版社：*アスキー・メディアワークス(2013-03-08)

定価：￥ 1,680

スクラムの父であるジェフ・サザーランドとケン・シュエイバーによる著者の日本語版。ビジネス層、マネジメント層向けにソフトウェア開発プロセス変革の必要性やアジャイル型開発プロセスの優位性について説明

How to Change the World ～チェンジ・マネジメント3.0～

*著者／訳者：*Jurgen Appelo, 前川哲次(翻訳), 川口恭伸(翻訳), 吉羽龍太郎(翻訳)

*出版社：*達人出版会

*定価：*500円

どうすれば自分たちの組織を変えられるだろう？それには、組織に変革を起こすチェンジ・マネジメントを学習することだ。アジャイルな組織でのマネージャーの役割を説いた『Management 3.0』の著者がコンパクトにまとめた変化のためのガイドブック

寄稿

WEB+DB PRESS Vol.100

*著者／訳者：*河原一哉, 白土慧, 菊田洸ほか

*出版社：*技術評論社(2017-8-24)

定価：￥ 1,598

100号記念選書、TOPエンジニアを支える一冊の企画に寄稿しました。紹介したのはアジャイル開発の原点の一つとも言える大野耐一著『トヨタ生産方式』です

Elastic Leadership

*著者／訳者：*Roy Osherove著, 島田浩二翻訳

*出版社：*オライリージャパン(2017-5-13)

定価：￥ 3,024

本書は日本語版オリジナルとして巻末に日本人エンジニアによるエッセイが14本収録されています。そのうち、42章：採用プロセスについてもっと考えよう、を当方が寄稿しました。リーダーシップが必要なのであれば採用のときにも候補者にリーダーシップがあるのか確認すべきですよね

WEB+DB PRESS Vol.93

*著者／訳者：*原田騎郎, 吉羽龍太郎, 松浦隼人, 須藤涼介ほか

*出版社：*技術評論社(2016-6-24)

定価：￥ 1,598

特集の「実践見積り」を寄稿。見積りでは、スケジュールやスコープ、メンバーのスキルなどの要素が密接に絡み合っていて難しいと思う人が多いと思います。本特集では見積りの歴史的経緯、従来型とアジャイル開発における見積りのやり方、見積りの今後について解説しています

WEB+DB PRESS Vol.83

*著者／訳者：*原田騎郎, 吉羽龍太郎, 山口陽平ほか

*出版社：*技術評論社(2014-10-24)

定価：￥ 1,598

特集の「強いチームの作り方」を寄稿。刻々とニーズが変化する近年のソフトウェア開発では、既成観念にとらわれない新しいアイデアが不可欠ですが一方で意見の対立がメンバーの衝突を引き起こすことも多くあります。本特集では、意見の違いをメンバーどうしが受け入れ、新しいアイデアへつなげていけるチームの作り方を解説しています

実践 Vagrant

*著者／訳者：*Mitchell Hashimoto

*出版社：*オライリージャパン(2014-02-21)

定価：￥ 2,808

Vagrantの概要やマシン設定から、プロビジョニング、ネットワーク、ボックス、さらにはプラグインによる拡張までを網羅した開発者自身による解説書。日本語版限定で「Vagrantプラグイン」と「Packer」について寄稿。また伊藤直也さんによる描きおろしも含まれている

100人のプロが選んだソフトウェア開発の名著 君のために選んだ1冊

*出版社：*翔泳社(2012-02-22)

定価：￥ 1,944

ソフトウェア技術者のための名著案内。国内最大級の開発者向けカンファレンス「DevelopersSummit」のベストスピーカーを中心とした、100名の執筆陣が書籍を選定・紹介。アジャイルプラクティスについて寄稿

Latest post:

- 【翻訳】スクラムは抽象クラス
- スクラムで削除された5つのトピック
- KeycloakをつかってAWSにSSO接続する方法
- 【書籍紹介】Fifty Quick Ideas To Improve Your User Stories
- 【新刊発売のお知らせ】業務システム クラウド移行の定石
- 【資料公開】Effective DevOps
- プロダクトオーナーのアンチパターン
- スクラムの概要を1分で理解できるイラスト【2018版】
- リリーススプリントとはなにか
- アジャイルコーチはなぜ1週間スプリントを勧めるのか
- スクラムにおける技術的スパイクの進め方
- 【資料公開】スクラムプロジェクト開始のベストプラクティス

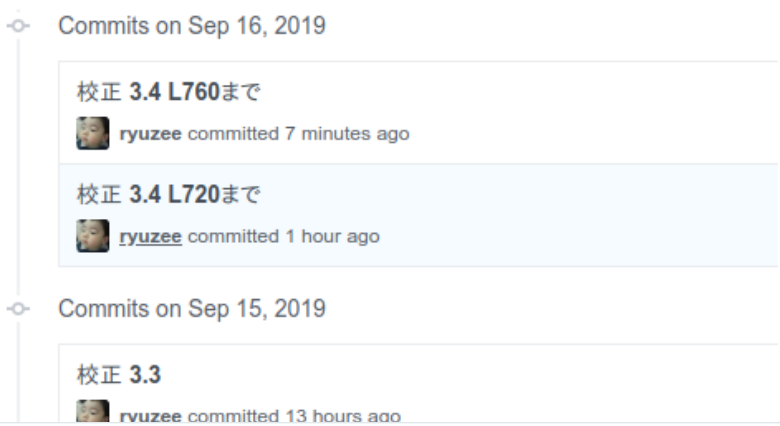
- アジャイル開発やスクラムのトレーニングでよく聞かれる質問とその答え (2)
- アジャイル開発やスクラムのトレーニングでよく聞かれる質問とその答え (1)
- Azure Cloud Shellを活用してスライド公開環境を30分で作る方法
- スクラムガイド2017での変更点の紹介
- スクラムマスターの仕事にはどんなものがあるか
- ふりかえりの実施状況に関する調査結果 (Annual Agile Retrospective Report)
- スクラムで開発チームが自由な取り組みをするには？
- スクラムの導入状況のサマリー (The State of Scrum Report 2017)

Tweets by @ryuzee

Ryutaro YOSHIBA
@ryuzee
早く印税だけで生きていけるようになりたい(嘘)

6h

Ryutaro YOSHIBA
@ryuzee
ひたすら原稿と向き合うマン



Embed View on Twitter

Archive

- 2018
- 2017
- 2016
- 2015
- 2014
- 2013
- 2012
- 2011
- 2010
- 2009
- 2000

Latest post

- 【翻訳】スクラムは抽象クラス(2018/09/20)
- スクラムで削除された5つのトピック(2018/08/04)
- Keycloakを使ってAWSにSSO接続する方法(2018/07/27)
- 【書籍紹介】Fifty Quick Ideas To Improve Your User Stories(2018/07/18)
- 【新刊発売のお知らせ】業務システム クラウド移行の定石(2018/07/11)
- 【資料公開】Effective DevOps(2018/04/25)
- プロダクトオーナーのアンチパターン(2018/03/12)
- スクラムの概要を1分で理解できるイラスト【2018版】(2018/03/08)
- リリーススプリントとはなにか(2018/02/26)
- アジャイルコーチはなぜ1週間スプリントを勧めるのか(2018/02/15)
- スクラムにおける技術的スパイクの進め方(2018/02/04)
- 【資料公開】スクラムプロジェクト開始のベストプラクティス(2018/01/11)
- アジャイル開発やスクラムのトレーニングでよく聞かれる質問とその答え (2)(2017/12/12)
- アジャイル開発やスクラムのトレーニングでよく聞かれる質問とその答え (1)(2017/12/05)
- Azure Cloud Shellを活用してスライド公開環境を30分で作る方法(2017/12/02)
- スクラムガイド2017での変更点の紹介(2017/11/09)
- スクラムマスターの仕事にはどんなものがあるか(2017/10/15)
- ふりかえりの実施状況に関する調査結果 (Annual Agile Retrospective Report)(2017/08/10)
- スクラムで開発チームが自由な取り組みをするには？(2017/08/06)
- スクラムの導入状況のサマリー (The State of Scrum Report 2017)(2017/08/04)

Tags

2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 agile alm ami api autoscaling aws azure bdd bootstrap cakephp capistrano chef cloud cloudformation coreos dashing deploy design devlove devops diary docker done doneの定義 eip electron fixture git grafana graphite hudson javascript jenkins linux lxc microservices middleman migration movie mysql nagios node.js oss packer php presentation publication rails ready recommended retrospective ruby sahara scrum selenium sensu serverspec slack slide slidehub subversion tdd test tps trac vagrant xp ふりかえり アジャイル アンチパターン オフショア カイゼン カンバン キャリア クラウド グルーミング コミット コミットメント コンサルタント コーチ コードレビュー スクラム スクラムオブスクラム スクラムマスター スコープ ストーリー ストーリーポイント スパイク スプリント スプリント0 スプリントバックログ スプリントプランニング スプリントレビュー タスク チーム テスト デイリースクラム デプロイ バグ バーンダウンチャート フィードバック プランニングポーカー プレゼンテーション プロジェクト プロダクトオーナー プロダクトバックログ ベロシティ メトリクス メール ユーザーストーリー ライフハック リファインメント リモートワーク リリース リリーススプリント リリースプランニング リーダー リーン レトロスペクティブ レビュー ロール ワーキングアグリーメント 可視化 契約 寿司 導入 技術的負債 採用 書籍 書評 本 検索 欠陥 監視 管理職 組織 継続的インテグレーション 継続的デプロイ 継続的デリバリー 自動化 見える化 見積り 評価