Documenter le déploiement

D'une application web (site web, Progressive Web App)

CONTENU

Description de la compétence 1
Introduction 1
Déploiement : Kesako ? 2
Déploiement manuel d'un site Web 3
Livraison & Déploiement automatique 4
L'intégration Continue (CI) 5
Le Déploiement Continu (CD) 5
Infrastructure as Code (IaC) 6
IaC : L'approche procédurale 6
IaC : L'approche déclarative 6

DESCRIPTION DE LA COMPETENCE

1. En tenant compte des dépendances et des versions de l'application, rédiger ou mettre à jour la procédure de déploiement de l'application.
2. Ecrire et documenter les scripts de déploiement.
3. Définir les environnements de tests pour les tests d'intégration, système et d'acceptation client
4. Réaliser une veille technologique sur les évolutions techniques et les problématiques de sécurité liées au déploiement d'une application web ou web mobile, y compris dans le cadre d'une démarche DevOps.

INTRODUCTION

Dans le cadre du développement d'une application (Application lourde, Site web, Progressive Web App…), il est nécessaire de bien documenter la procédure de déploiement de l'application.

Cette documentation inclura systématiquement les informations sur l'environnement d'exécution de l'application, la procédure d'installation de l'application (avec les dépendances) et les procédures de tests.

De plus, A chaque nouvelle version (correction de bug, ajout de fonctionnalité…), la procédure de déploiement de la mise à jour devra également être rédigée.

Lors de chaque déploiement ; l'équipe DevOps aura la charge de tester l'application en conditions réelles, la communication avec les DevOps doit donc être claire et sans ambiguïté !

DEPLOIEMENT : KESAKO ?

Déployer une application ou un site web signifie « appliquer un procédé permettant d'installer ou mettre à jour le site web sur un environnement donné ».

Prenons un exemple pratique :

Vous avez à développer un site web, mais le client n'a pas encore son hébergement. Le développement va donc se faire sur votre machine locale pour le développement. Une fois terminé, les fichiers du site web seront transférés sur un serveur que vous possédez pour pouvoir tester le site « en conditions réelles », et permettre au client de le tester pour ses derniers retours. Enfin, après validation, tout le dossier sera transféré sur l'hébergement du client pour la mise en production.

Dans cet exemple :

- **L'environnement de développement** est votre machine locale
- **L'environnement de test** est votre serveur (ou un serveur de votre entreprise)
- **L'environnement de préproduction** est votre serveur (ou un serveur de votre entreprise)
- **L'environnement de production** est le serveur ou l'hébergement du client.

Les versions des applicatifs serveur (serveur HTTP, SGBD…) doivent être identiques sur tous les environnements pour s'assurer que l'application se comportera de la même manière sur chacun d'entre eux.

Le processus de déploiement pourrait donc être le suivant : se connecter via SFTP à un serveur dépendant de l'environnement, puis déplacer les fichiers de l'environnement de développement vers le serveur, dans le bon dossier. Enfin, vérifier que le site est accessible via le réseau (sur une adresse qui dépend de l'environnement) et fonctionne comme attendu.

Dans l'exemple ci-dessus, la connexion, le transfert et les tests sont faits à la main, nous appellerons donc ce processus de déploiement "*Déploiement manuel via SFTP*".

Ce processus de déploiement, en apparence très simple, impose 2 règles majeures qu'il faut respecter :

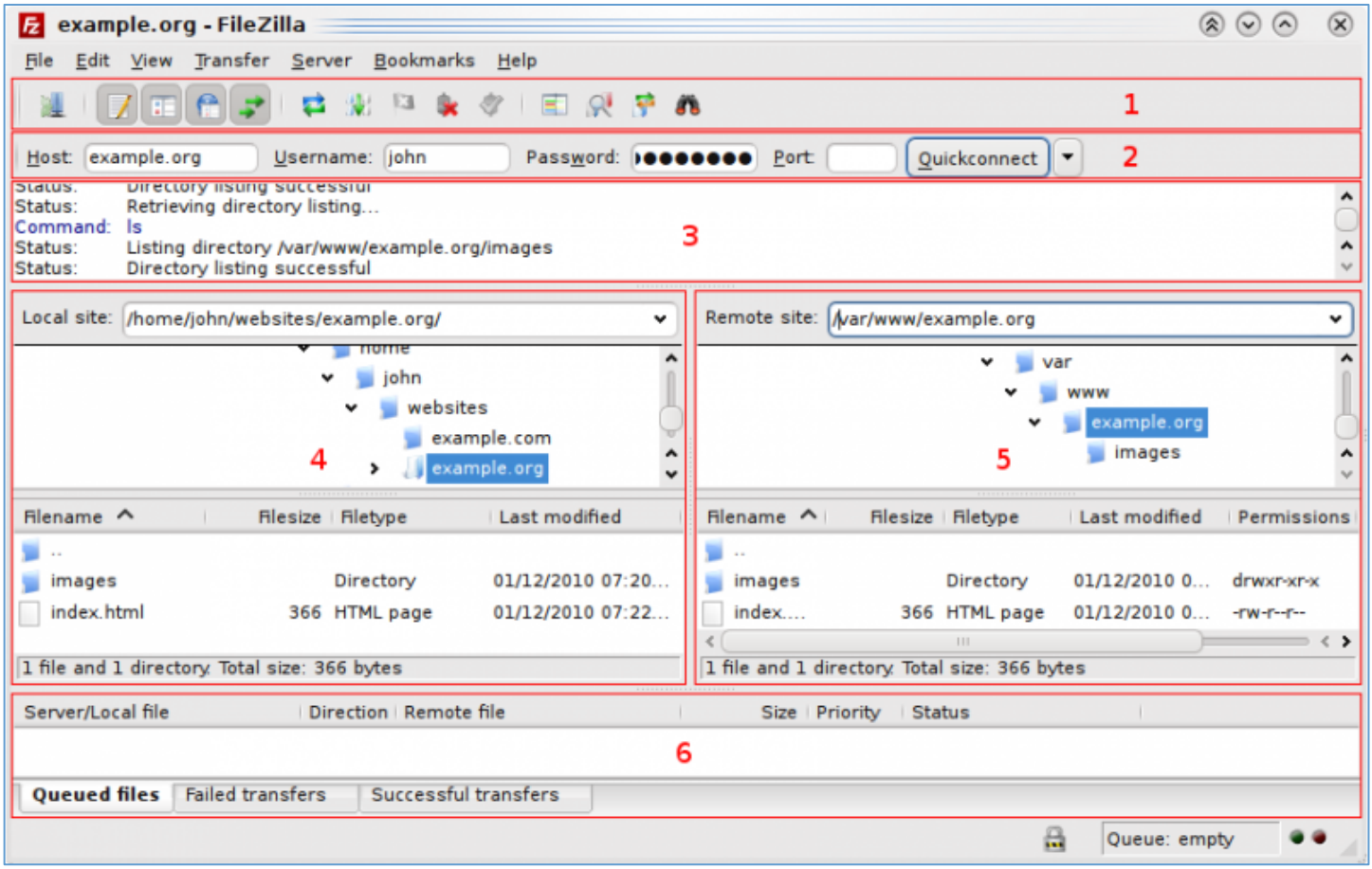
1. Les fichiers ne peuvent aller que dans un sens, de l'environnement de développement vers l'environnement cible.
2. Seuls les fichiers de l'environnement de développement doivent être modifiés. Les éditions « à l'arrache » sur l'environnement de production sont une violation des bonnes pratiques et doivent être bannies de vos procédures de travail.

DEPLOIEMENT MANUEL D'UN SITE WEB

Avant que se généralise les démarches de déploiement continu et automatique, les équipes de développement procédaient à un déploiement manuel des applications et sites web. Dans un tel cas, la documentation détaille les opérations à réaliser pour déployer/installer l'application.

Les éléments à documenter sont les suivants (liste non exhaustive) :

- Le ou les langages utilisés dans le projet (avec leur numéro de version)
- Les éventuelles dépendances nécessaires au projet (avec leur numéro de version)
- Le ou les applicatifs serveur utilisés (avec leur numéro de version)
- Les détails sur la configuration de chacun des éléments précédents
- Les informations concernant chaque environnement
    - L'adresse du serveur pour le transfert des fichiers via SFTP
    - L'adresse du serveur de base de données
    - Le chemin où doivent être transférés les fichiers du projet
- L'adresse (nom de domaine ou IP) pour accéder à l'application



Capture d'écran d'un logiciel de transfert de fichiers pour les applications Web

LIVRAISON & DEPLOIEMENT AUTOMATIQUE

Depuis quelques années, notamment depuis la généralisation des démarches type « DevOps », les projets sont configurés pour utiliser des outils permettant de compiler et/ou déployer automatiquement une application lorsqu'une mise à jour du code est effectuée sur une branche définie du référentiel du projet. Généralement, il s'agit de la branche principale du référentiel.

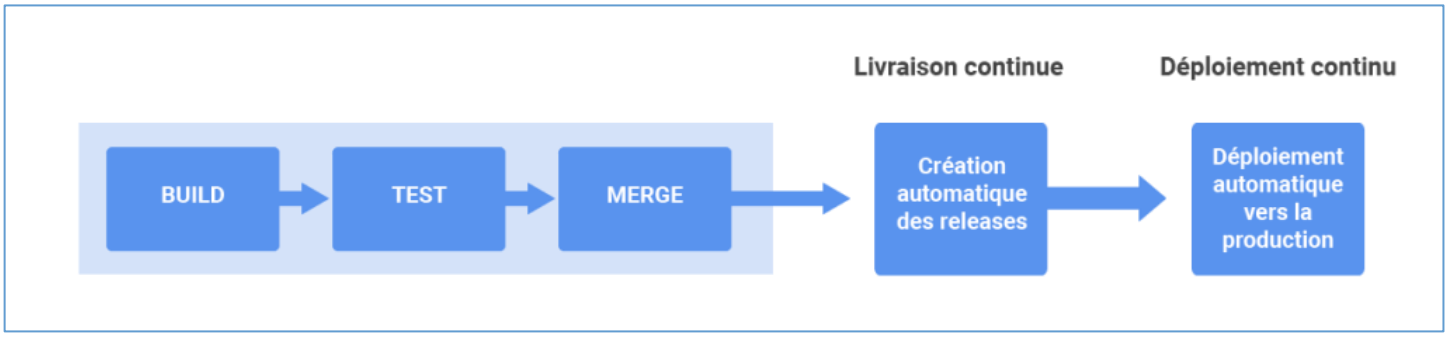
Qui dit branche, dit système de gestion de version (GIT, SVN…).

Dans un tel scénario, les développeurs travaillent sur des branches distinctes. Leur travail est ensuite fusionné dans une branche de développement. Lorsque le code de la branche de développement est prêt à être mis en production, il est fusionné dans la branche principale.

Évènement sur le référentiel		Actions réalisées automatiquement
Code fusionné sur la branche de développement	→	1 - Déclenchement des tests automatisés 2 - Déploiement sur le serveur de test
Code fusionné une branche "release"	→	1 - Déclenchement des tests automatisés 2 - Déploiement sur le serveur de préproduction
Code fusionné sur la branche principale	→	1 - Déclenchement des tests automatisés 2 - Déploiement sur le serveur de production

De la même manière, lorsqu'il est nécessaire de faire évoluer plusieurs versions d'un même logiciel (la V2 est déployée tandis que la V1 bénéficie encore de mises à jour), il y aura une branche « principale » qui héberge le code de la dernière version et des branches secondaires qui seront associés à un numéro de version spécifique. Cette organisation demandera plus de rigueur aux équipes car certains « commit » seront à fusionner avec la branche principale ou sur une des branches correspondant à une version spécifique tandis que d'autres seront fusionnés sur plusieurs branches (cas d'une mise à jour de sécurité par exemple qui sera déployée sur la V1 et la V2).

Ces principes d'automatisation sont le point de départ de l'industrialisation des processus permettant l'intégration et le déploiement continu (CI/CD) d'une application.



DevOps est toujours associé au mot « continu » et est toujours synonyme de « livraison et/ou déploiement » rapide du code produit vers les environnements de production.

L'INTEGRATION CONTINUE (CI)

Sigle	Anglais	Français
CI	Continuous Integration	Intégration Continue

La **CI** désigne l'intégration continue, à savoir un processus d'automatisation pour les développeurs. Cette intégration continue consiste, pour les développeurs, à apporter régulièrement des modifications au code de leur application, à les tester, puis à les fusionner dans un référentiel partagé. Cette solution permet de fluidifier le processus de développement mais aussi d'éviter les régressions.

La CI permet d'automatiser les tests à des fins qualitatives, afin de produire des « release » qui seront déployées avec les principes de « déploiement continu ».

LE DEPLOIEMENT CONTINU (CD)

Sigle	Anglais	Français
CD	Continuous Delivery	Déploiement Continu (ou Distribution Continue)

La **CD** peut désigner la « distribution continue » ou le « déploiement continu », deux concepts très proches, parfois utilisés de façon interchangeable.

La CD vous permettra de déployer automatiquement vos releases sur vos environnements de test puis de production.

INFRASTRUCTURE AS CODE (IAC)

L'infrastructure as Code (IaC) est une méthodologie qui permet de gérer l'automatisation de l'infrastructure cible. Toute la configuration de l'environnement d'exécution de votre application se trouve dans le référentiel partagé (GIT).

Dans cette méthodologie, nous pouvons utiliser 2 approches :

- L'approche procédurale
- L'approche déclarative

IAC : L'APPROCHE PROCEDURALE

L'approche procédurale revient à se poser la question « Comment l'infrastructure devrait être changée ? »

C'est une méthode itérative dans laquelle les développeurs mettent en place différents scripts d'automatisation afin de consolider l'infrastructure.

IAC : L'APPROCHE DECLARATIVE

L'approche déclarative consiste à définir l'infrastructure dans des fichiers de configurations. Un outil « IaC » est ensuite utilisé pour configurer l'infrastructure selon la configuration déployée sur le référentiel (les fichiers de configuration étant stockés dans un système de contrôle de code source (GIT).

--- FIN DU DOCUMENT ---