IOS ACADEMIA PROGRAMMING SCHOOL

【Git】概念説明と開発現場での必要性

はじめに

この記事では、Gitについての概念や開発現場における必要性について説明しています。
そもそもGitとは何なのか。また、何故それが必要なのかを理解することで、納得感をもってGitについて学んでいきましょう。

Gitとは？

Gitとは、バージョン管理システム(「VSC(Version Control System)」とも呼ばれるもの)のことです。
一般的によく耳にする「バージョン」と言えば、アプリのバージョンやゲームなどのバージョン(v1.2.0など)が思い浮かぶかと思います。

Gitが管理するバージョンも、基本的な意味としてはこれらと同じものだと思っていて問題ありません。(※1)
そして、Gitでは何のバージョンを管理するかと言うと、エンジニアが開発をする「ソースコード」がその対象となります。

エンジニアは、1人または複数人でソースコードを書き換えることによって、ソフトウェアのアップデートを行いますが、その「書き換え」の履歴を管理することをGitの世界では「ソースコードのバージョンを管理する」という様に考えます。

つまりGitとは、**「ソースコードのバージョン管理ツール」** であると言えます。

開発現場での必要性

では、開発現場では何故Gitを使用してソースコードのバージョン管理をする必要があるのでしょうか？
それは、バージョンを管理することによって、開発時(≒ソースコードの書き換え時)により効率よく作業を進めることができる、という恩恵があるためです。

これは1人で開発を行っている場合と、複数人で開発を行っている場合とでそれぞれ異なる恩恵があります。
では実際に「Gitが無い場合」と「Gitがある場合」とで、それぞれの具体的なケースを見ていきましょう。

1人で開発している時

例えばAさんが個人アプリを1人で開発していると仮定します。
毎日コツコツと自分のアプリのソースコードの書き換えを行っていき、ようやく新しい機能の実装が終えられました。

そして、アプリをリリースする前に念の為に動作確認をしてみたところ、正常に動作しない機能が見つかってしまいました。
これはいわゆる「バグ(不具合)」が発生してしまっている状態です。

しかし開発中、最後に正常動作していることを確認したのは10日前のことでした。
そのため、10日前から今までの「どのタイミングでバグが発生したのか」検討もつきません。

Gitが無い場合

仕方がないので、Aさんは **「10日前から今日までに改修した全範囲」** を調査してバグを探すことにしました。

Gitがある場合

AさんはGitを使って毎日の作業内容を1日単位で区切ってバージョン履歴を残していました。

そこでAさんは、まず5日前時点の状態にソースコードを巻き戻して動作確認をしてみたところ、ここでは正常に動作していることが確認出来ました。
つまり、5日前時点ではバグが発生していなかったと分かります。

次に2日前時点の状態にソースコードを巻き戻して動作確認をしてところ、これも正常動作することが確認出来たので、少なくとも2日前時点まではバグは無さそうです。

では同じ様に1日前時点の状態にソースコードを巻き戻してみたところ、正常動作しなくなってしまいました。

このことから、バグは2日前から1日前の間に発生したことが分かったため、Aさんは **「2日前から1日前の間に改修した範囲」** を調査することにしました。

1人で開発している時にGitを使う恩恵

この例からは主に以下の様な恩恵があることが分かります。

- ソースコードの状態を任意の状態に巻き戻せる
- 不具合発生時などに調査対象範囲を限定できる
- バージョン毎の変更範囲を明確に把握できる

複数人で開発をしている時

では次に複数人で開発をしていると仮定しましょう。
AさんとBさんは同じアプリのソースコードに対してそれぞれ同時並行して開発を進めて行くことになりました。

二人はそれぞれ開発を終えて、リリースの準備をするために動作確認を行い、二人とも正常動作することを確認しました。

問題が無かったので、AさんとBさんのソースコードを合成して1つのソースコードにし、リリースの準備を進める事になりました。

Gitがない場合

1つのソースコードにするため、AさんのソースコードにBさんの開発した部分のソースコードを上書きする形で合成を行いました。

しかし、合成されたソースコードで念のため動作確認をしたところ、Aさんの改修したソースコードの一部が、Bさんのコードによって上書きされてしまったため、正常動作しないことが分かりました。

Aさんは上書きによって消されてしまったソースコードの内容を思い出しながら、ソースコードを修正することにしましたが、ここでも問題が。

Bさんによって書き換えられたソースコードの部分については、Aさんは把握していないため、Bさんに改修内容を聞きながら適切に修正行う必要がありそうです。

こうしてリリース前に二人は一度時間をとって、どの部分をどう直さなければいけないのか、**Aさんの記憶とBさんの実装説明を元に修正**をすることになりました。

Gitがある場合

Gitを使用して2つ以上のソースコードを合成する時、同一箇所が修正されていた場合にはそもそも合成が出来ません。

その場合は、同一修正箇所がGitによって洗い出されるため、**Aさんはその箇所だけに対して、Bさんの修正内容を破壊しないように適切に修正**を加えた後に合成を行いました。

同一修正箇所を合成前に適切に対処したことで、合成後も正常動作することを確認でき、リリースを行いました。

複数人で開発をしている時にGitを使う恩恵

この例からは主に以下の様な恩恵があることが分かります。

- 複数人で同時開発中に競合(同一修正)箇所があった場合、合成前に問題を確認して適切に対処ができる
- 合成後のソースコードに対して意図しない競合が発生することを抑制でき、安全性が保たれる

まとめ

この様に、個人開発であっても複数人によるチーム開発であっても、Gitを使用することによってさまざまな恩恵を受けることができます。ここで紹介した以外にもGitを使うことによる恩恵はたくさんあるので、使い方を学習していきながら少しずつ理解を深めていきましょう。

注釈

※1 : 厳密には「v1.2.0」などは[セマンティックバージョニング]という考え方に沿った形式なので、Gitの管理方法とは異なる部分があります。

← 教材一覧へ戻る