**O que é o Maven?**

O Maven é uma ferramenta de integração de projetos, e é responsável por gerenciar dependências, controlar versão de artefatos, gerar relatórios de produtividade, garantir execução de testes, manter nível de qualidade do código dentre outras.

Com o Maven consiguimos isolar as bibliotecas usadas no projeto em um ‘repositório‘ compartilhado pela equipe, ou por toda internet no caso do repositório central do Maven. Dessa forma não nos preocupamos com duplicidade de dependências entre módulos do projeto e nem em disponibilidade das mesmas no repositório de código. Quanto a versão das dependências, estas ficam centralizadas em arquivos de configuração dos projetos de forma explícita e hierarquisada pelos módulos. Com isso o Maven consegue se encarregar de fazer as devidas substituições de bibliotecas e identificar possíveis falhas de dependências.

**Benefícios**

**Modelo de Objetos de Projetos (POM)**

É um arquivo em formato .xml, que define as informações necessárias para que o Maven possa executar um conjunto de metas e que a construção do software possa ser realizada.

O pom.xml contém uma lista de arquivos de configuração, uma lista de desenvolvedores que atuam em papéis, um sistema de controle de ocorrências, informações sobre a empresa e licenças, URL do projeto, dependências e etc.

**Repositório**

O Maven utiliza dois tipos de repositórios para armazenar os artefatos: o local e o remoto.

**Repositório local:** O repositório local armazena os artefatos localmente no seu computador. É criado um diretório .m2 no seu diretório de usuário. Os artefatos locais são acessados rapidamente e são consultados sempre que o Maven precisa resolver uma dependência ou executar um plug-in.

**Repositório remoto:** O repositório remoto pode estar em um dos vários repositórios Maven espalhados pela internet ou mesmo na sua intranet. Vários projetos de código livre publicam suas bibliotecas nestes repositórios remotos. Assim, se seu projeto utilizará algum framework de código livre comumente utilizado (como Hibernate ou Spring).

Baseando-se na informação de dependência descrita no POM, o Maven tentará resolver as dependências na seguinte ordem:

A dependência é procurada no repositório local.

A dependência é procurada em todos os repositórios remotos configurados.

Falhando 1 e 2, um erro é reportado.

**O ciclo de vida do processo build**

Clico de vida e fases: Controlam o processo de produção.

Ciclo de vida de construção de fases: associado ao POM que detalha todas as dependências de arquivos e artefatos necessários criando assim uma rotina para cada vez que o código for requisitado baixando ou agregando tudo o que for necessário para o seu funcionamento como um plug-in que é descrito no POM verificando versões e requisitando arquivos por todo ciclo de vida do projeto, provendo o reúso dos artefatos de forma da rotina de leitura do arquivo POM, o qual vai estar descrito tudo o que é necessário, verificação de alterações e integração de dependências.

Fases do ciclo de vida:

* Validate
* Initialize
* Generate Sources
* Process Sources
* Generate Resources
* Process Resources
* Compile
* Process Planes
* Generate Test Sources
* Process Test Sources
* Generate Test Resources
* Test Compile
* Test
* Package
* Pre-integration Test
* Verify
* Install
* Deploy

**Principais conceitos básicos**

**Gerenciamento de Dependências:** O maven centraliza as suas dependências de uma forma que você não precise ficar sempre procurando na internet, etc para tê-las em seu aplicativo. Basta declarar no arquivo pom.xml do Maven quais as libs você vai precisar (e em qual o repositório delas) para que ele sempre baixe e gerencie. Isso facilita na hora de trabalhar em equipe, por exemplo, onde você teria que ficar copiando o projeto enorme para o colega. Com o Maven você passa apenas o projeto com o pom.xml e lá no outro PC ele baixa as mesmas dependências.

**Controle de build:** Por exemplo, se você quiser que o build seja feito se e somente se todos os casos de testes de sua aplicação derem certo. Isto é configurável no maven. Outro caso seria realizar a mitificação e ofuscação do seu código, o maven tem plugins para isso, de forma que você não precise fazer manual, ele faz na hora da build. Além disso ele permite que você controle toda a fase de build, caso queira executar algo diferente antes ou após determinada fase.

**Arquétipos:** É comum projetos parecidos terem dependências parecidas, por exemplo, projetos Web com JSF + Hibernate + etc. Para não ficar sempre criando um novo projeto do zero, configurando tudo , o Maven possui o conceito de Arquétipos, que são espécie de "esqueletos prontos" onde já vem tudo configurado para você. Isso traz um ganho enorme de tempo na inicialização/configuração de um novo projeto.

**Plugins:** Para o Maven existem também diversos plugins, onde você pode utilizar determinado recurso através dele. Por exemplo, o JSLint que verifica a sintaxe de seus javascripts.