yaml에 동적으로 변수 사용하기

현재 진행 중인 프로젝트는 이전 위키북 프로젝트를 진행할 때 만든 빌드, 배포 스크립트를 사용하고 있다. 이 빌드, 배포 스크립트는 온라인에서 EP로 활동하고 있는 박응주라는 친구가 만들었다. 유용한 부분이 많아 일부 개선해 가면서 계속해서 사용하고 있다.

이 빌드, 배포 스크립트는 설정 파일로 yaml을 사용하고 있다. 2년 전 처음 프로젝트를 시작할 때는 2, 3개의 설정으로 충분했다.

```
applications:
    slipp-web:
        type: WebApp
        scm: svn//www.javajigi.net
        src_dir: /build-projects/integration-workspace/slipp
        compile: mvn -U -Pintegration clean install
        build: mvn -U -Dmaven.test.skip=true -Pintegration -pl web -am clean install
        catalina_base: /web-projects/slipp-web
        catalina_home: /usr/apps/apache-tomcat-6.0.35
        java_opts: -Xms128m -Xmx512m -XX:MaxPermSize=256m
        rsync: rsync -avr --delete /build-projects/integration-workspace/slipp/web/webapp/ /web-projects/slipp-
web/webapps/
```

그런데 2년 동안 프로젝트를 진행하면서 빌드, 배포해야 되는 서비스는 점점 더 많아졌다. 최근에는 거의 10개에 달하는 서비스를 각각 빌드해야 한다. 서비스가 늘어나면서 이 yaml 설정도 증가했다. 설정이 증가하면서 발생하는 이슈는 뭐니 뭐니 해도 중복이었다.

```
applications:
    slipp-web:
        type: WebApp
        scm: svn//www.javajigi.net
        src_dir: /build-projects/integration-workspace/slipp
        compile: mvn -U -Pintegration clean install
        build: mvn -U -Dmaven.test.skip=true -Pintegration -pl web -am clean install
        catalina_base: /web-projects/slipp-web
        catalina_home: /usr/apps/apache-tomcat-6.0.35
        java_opts: -Xms128m -Xmx512m -XX:MaxPermSize=256m
        rsync: rsync -avr --delete /build-projects/integration-workspace/slipp/web/webapp/ /web-projects/slipp-
web/webapps/
    slipp-admin-web:
        type: WebApp
        scm: svn//www.javajigi.net
        src_dir: /build-projects/integration-workspace/slipp
        compile: mvn -U -Pintegration clean install
        build: mvn -U -Dmaven.test.skip=true -Pintegration -pl web -am clean install
        catalina_base: /web-projects/slipp-admin-web
        catalina_home: /usr/apps/apache-tomcat-6.0.35
        java_opts: -Xms128m -Xmx512m -XX:MaxPermSize=256m
        rsync: rsync -avr --delete /build-projects/integration-workspace/slipp/web/webapp/ /web-projects/slipp-
admin-web/webapps/
    slipp-mobile-web:
        type: WebApp
        scm: svn//www.javajigi.net
        src_dir: /build-projects/integration-workspace/slipp
        compile: mvn -U -Pintegration clean install
        build: mvn -U -Dmaven.test.skip=true -Pintegration -pl web -am clean install
        catalina_base: /web-projects/slipp-mobile-web
        catalina_home: /usr/apps/apache-tomcat-6.0.35
        java_opts: -Xms128m -Xmx512m -XX:MaxPermSize=256m
        rsync: rsync -avr --delete /build-projects/integration-workspace/slipp/web/webapp/ /web-projects/slipp-
mobile-web/webapps/
```

위 설정에서 볼 수 있듯이 대부분의 설정은 똑같다. 각 서비스에 따라 달라지는 부분은 서비스명만 달라질 뿐이다. yaml을 통해 이 이슈를 해결하려고 했으나 yaml은 한계가 있었다. http://yaml.org/spec/1.2/spec.html#id2786196 문서를 참조해 &,*를 통해 앵커를 지정하고 사용하는 방법을 사용하려고 했으나 내가 원하는 기능은 아니었다. 왜 한계가 있었는지는 yaml의 &, *를 직접 사용해 보면 알 수 있다.

그래서 해결책을 찾기 위한 검색 시작. 찾다 보니 나와 비슷한 고민을 한 사람들이 있어 다음과 같이 해결했다. 현재 빌드, 배포 스크립트는 ruby 기반으로 구현되어 있다. 먼저 yaml에 동적인 부분을 별도의 설정으로 분리해 관리할 수 있도록 설계했다.

```
properties:
    src_dir: /build-projects/integration-workspace/slipp
    deploy_dir_prefix: /web-projects
    catalina_home: /usr/apps/apache-tomcat-6.0.35
applications:
    <%= app_name %>:
        type: WebApp
        scm: svn//www.javajigi.net
        src_dir: <%= src_dir %>
        compile: mvn -U -Pintegration clean install
        build: mvn -U -Dmaven.test.skip=true -Pintegration -pl web -am clean install
        catalina_base: <%= deploy_dir_prefix %>/<%= app_name %>
        catalina_home: <%= catalina_home %>
        java_opts: -Xms128m -Xmx512m -XX:MaxPermSize=256m
        rsync: rsync -avr --delete <%= src_dir %>/web/webapp/ <%= deploy_dir_prefix %>/<%= app_name %>/webapps/
```

중복이 발생하는 부분을 먼저 properties로 모두 추출했다. 그리고 중복을 제거하다보니 앞의 소스 코드와 같이 3개의 설정이 필요한 것이 아니라 하나의 설정으로 모두 해결할 수 있었다. 이 같은 방식으로 진행하면 앞으로는 서비스가 추가되더라도 별도의 설정을 추가할 필요가 없게 되었다.

그렇다면 위와 같이 설정한 yaml은 어떻게 처리했을까? 소스 코드를 살펴보면 다음과 같다.

```
require 'yaml'
require 'erb'
require 'ostruct'

config = YAML::load_file(ENV['config'])
properties = config['properties']
if properties != nil
        properties['app_name'] = ENV['app']
        renderer = Renderer.new(properties)
        result = renderer.render(renderer.getTemplate(ENV['config']))
        config = YAML::load(result)
end
apps = config['applications']
class Renderer < OpenStruct
        def getTemplate(fileName)
                File.open(fileName, "r") do |f|
                    ERB.new(f.read())
            end
        end

        def render(template)
            template.result(binding)
        end
end
```

먼저 위 설정 파일을 읽는다. 설정 파일을 읽은 후에 properties 값이 있을 경우 ruby의 erb 템플릿 기능을 활용해 properties 값을 설정 파일에 전달해 새로운 설정 파일을 생성해 낸다(renderer.render(renderer.getTemplate(ENV['config']))).

erb를 통해 새로 생성한 설정 파일을 yaml을 모듈을 활용해 다시 로드한 후 빌드를 하는 시점에 사용한다. 약간 무식한 방법일 수도 있지만 현재로서는 달리 다른 방법을 찾지 못해 위와 같이 해결했다. properties 설정 값이 존재할 때만 추가적인 작업 처리하도록 구현해 기존에 이미 존재하던 설정 파일을 그대로 활용할 수 있도록 했다.

해결 방법을 찾기 위해 오랜만에 잔머리 좀 굴렸더니 재미있게 일할 수 있었다. ruby는 많이 사용하지 않았는데 이 참에 다시 함 도전해 보고 싶다는 욕심도 생긴다. 앞으로 yaml에 동적인 변수를 사용하고 싶다면 ruby일 경우 erb 함 도전해 보면 좋겠다. 더 좋은 해결책이 있다면 더 좋고...

.