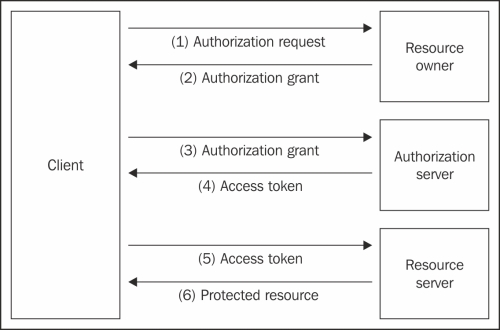
**http://bcho.tistory.com/m/942**

**OAuth 용어 정리**

Resource Owner (사용자)

Authorization Server (인증서버

Resource Server (REST API)



**OAuth 2.0 grant flow**

**Authorization code grant flow**

주로 Web Application에서 사용됨

**Implicit grant flow**

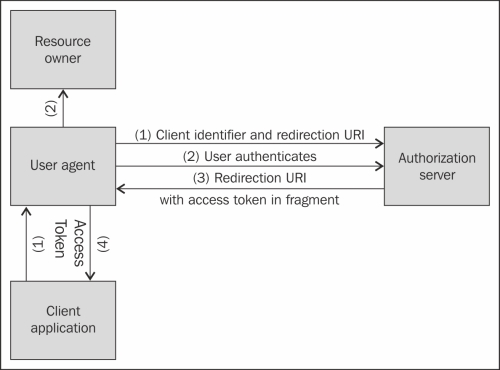
*자바스크립트* 애플리케이션에서 많이 사용됨. 스크립트 단에서는 credential 등이 노출 될 수 있으니, 주로 Read only 용도로 많이 사용함. accessToken이 노출될것을 전제로 함.

모바일 애플리케이션도 많이 사용하는걸로 나오네??

   Used in public clients

   It's is a redirection-based flow (similar to the one in the authorization code grant)

   The access token is received as a parameter of the redirection endpoint upon successful completion of the request, similar to the authorization code parameter in the authorization request response in the authorization code grant



Flow

①     The first step is initiation of the flow. The client redirects the User agent to the Authorization server by using the authorization endpoint, the client identifier, and the redirection endpoint that will be used for the response.

②     The Authorization server authenticates the Resource owner and requests his decision whether to authorize or deny the request.

③     If the Resource owner authorizes the request (which is assumed), he is redirected back with response information, using the supplied redirection endpoint that was provided with the initial request. The response information is contained in the URL fragment that contains the access token and other parameters (we'll see the difference between a regular URL parameter and one found in a URL fragment in the detailed overview).

④     Now that the User agent (the browser) is redirected back, the access token included in the response is passed to the Client application.

Client Server Application Case

Mobile App Case

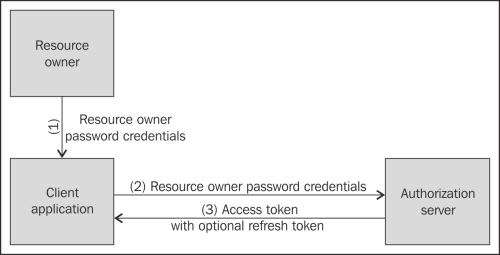
(웹이 아니기 때문에, Redirect 처리를 어떻게 해야 할지 고민해야 함.

Samsung Account와 같이 전용 APK를 넣는 방식이나, 웹 페이지 Scrapping 방식등이 있음)

**Resource owner password credential grant flow**

직접 ID,PASSWORD를 보내는 방식으로, 1’st level 파트너나 자사 시스템에 많이 사용.

기존의 HTTP BASIC이나 HTTP Digest 인증 방식을 migration하기가 용이함



①     The resource owner (for example, the user) supplies the Client application with his username and password.

②     The client application makes a request to the Authorization server, including the user's credentials and also his own identifier and secret.

③     The Authorization server authenticates the client based on his identifier and secret, checks whether it is authorized for making this request, and checks the resource owner credentials and other parameters supplied. If all checks pass successfully, the Authorization server returns an access token in response.

**Client credential grant flow**

Userless 상태에서 많이 사용됨. (API 키와 유사한 방식)

\* Authroization Code

- Redirect base / 서버 백엔드가 있는 경우 사용 - 파트너사가 API를 사용하는 시나리오에 유리

\* Implicit

- Rediect base / 특히 Java script 처럼 서버 백엔드가 없는 경우 유용. Read Only 등에 사용

\* Resource Owner Password Credential

- Client Id, Secret을 앱에 넣은 후, Client Id/Password로 인증하여, access token을 발급 받는 방식으로, Authorization Server와 Resource Owner가 같은 서비스인 경우 유용함 (자사 API 제공에 유용)

\* Client Crendetial

-

**유용한 Link**

* 서버 구현체 : http://oauth.net/2/
* http://aaronparecki.com/articles/2012/07/29/1/oauth2-simplified
* node.js 모듈 - https://github.com/jaredhanson/oauth2orize
* http://vinebrancho.wordpress.com/2014/05/19/oauth2-restapi-server-%EB%AA%A8%EB%B0%94%EC%9D%BC-%EC%95%B1%EC%9D%84-oauth2-0%EC%9C%BC%EB%A1%9C-%EC%A2%80-%EB%8D%94-%EC%95%88%EC%A0%84%ED%95%98%EA%B2%8C/