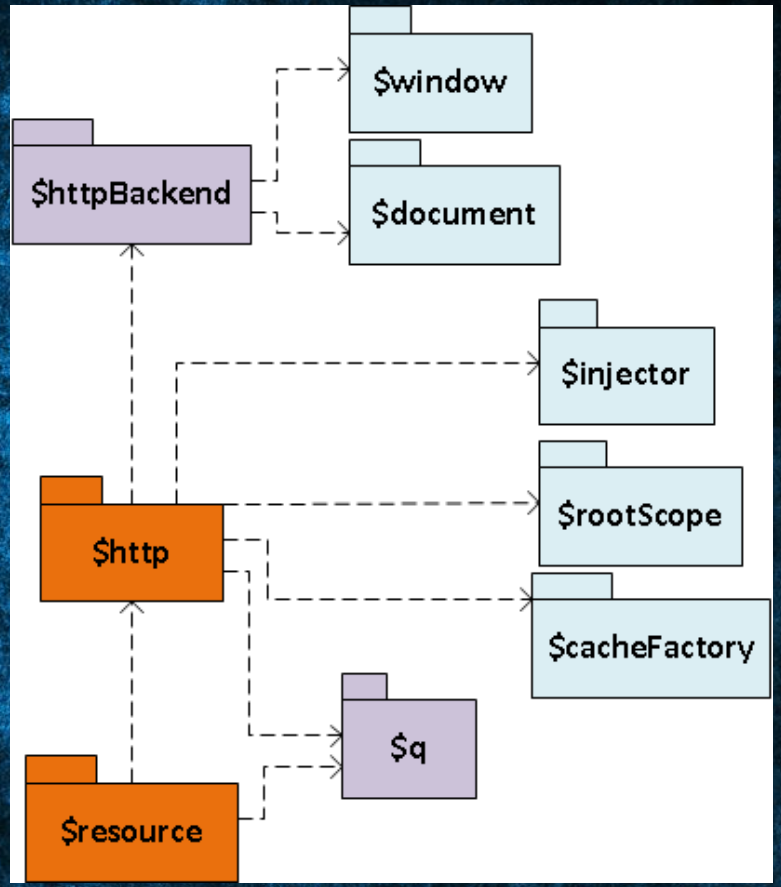
Services requêtes asynchrones

- 2 services différents
  - $http
    - Service le plus général
  - $resource
    - Mappeur RESTful
- Dépendances importantes
  - $httpBackend
  - $q



Promesses

- Différer une action = Faire une promesse
  - L'action finit par être effectuée => la promesse est « résolue »
  - Une erreur se produit => la promesse est « rejetée »
  - On peut à tout moment « notifier » la promesse (avancement dans la résolution de cette promesse)
- Toute requête asynchrone retourne une promesse
- Service $q
  - Création d'actions différées
    - var action = $q.defer();
  - Promesse sur chaque action différée
    - var promesse = action.promise;

Utiliser les promesses

- On enregistre des fonctions de rappel (*callback*) sur les promesses
  - promesse.then(callbackSucces, callbackErreur, callbackNotification)
- Pour résoudre/rejeter/notifier une promesse (et déclencher l'exécution des *callbacks*) :
  - action.resolve(résultat)
  - action.reject(raison)
  - action.notify(valeur)

Aggréger les promesses

- var promesse = $q.all([promesse1, promesse2, …]);
  - Renvoie une promesse qui sera résolue lorsque toutes les promesses passées dans le tableau seront résolues
    - Si l'une au moins produit une erreur, la promesse finale sera en erreur
- var promesse2 = promesse1.then(function(resultat) {
  return faisQqChose(resultat);
  });
  - Lorsque promesse1 est résolue, promesse2 est résolue également, son résultat étant celui retourné (dépend du résultat de promesse1).

Synchrone VS Asynchrone

- Synchrone :

```
try {
        action1();
        action2();
        console.log("ok");
} catch (erreur) {
        console.log("ko");
}
```

- Asynchrone :

```
var action1 = $q.defer();
var action2 = $q.defer();
var promesse = $q.all(
        action1.promise,
        action2.promise);
promesse.then(function(resultat) {
        console.log("ok");
}, function(erreur) {
        console.log("ko");
});
```