http://kodaktor.ru/json/users

```
{
    "users": [{
        "login": "student",
        "password": "tneduts"
    }, {
        "login": "myuser",
        "password": "mypas"
    }, {
        "login": "teacher",
        "password": "qq"
    }, {
        "login": "myking",
        "password": "myqueen"
    }]
}
```

Рассмотрим проектирование отображения **с** помощью шаблонов начнём с **JSON**

```
<div id="users">
  <ol>
    <li v-for="user in users">
       {{user.login}}
    </li>
  </ol>
</div>
```

Популярный простой механизм создания “вьюшек” **Vue (View)**

http://kodaktor.ru/jquery_vue

```
<!DOCTYPE html>
<html>
 <head>
   <title>jQuery + Vue</title><style> #users {display:none} </style>
   <script src="//ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>
   <script src="http://vuejs.org/js/vue.js"></script><script src="/j/out"></script>
   <script>
     $(()=>{
        //Out.console();
       $('h2').on('click',function(){
         $.ajax({
            dataType:'json',
            url:'/j/users',
            method:'GET'
         })
           .done(result=>{
              new Vue({
                  el:'#users',
                  data:{
                     users:result.users
                  }
              });
              $('#users').show('slow');
           })
           .fail(()=>console.log('Error!'))
           .always(()=>console.log('Fin!'));
       });
     });
  </script>
```

На первом месте здесь
императивный алгоритм
он управляет представлением

http://kodaktor.ru/vue_jquery

```
<head>
  <title>Vue + jQuery</title><style> #users {display:none} </style>
  <script src="//ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>
  <script src="http://vuejs.org/js/vue.js"></script><script src="/j/out"></script>
  <script>
    $(()=>{
      Out.console();
      $('h2').on('click',function(){
        let el = '#users';
        new Vue({
             el:el,
             data:{
                 users:''
             },
             created: function() {
                 this.fetchData()
             },
             methods: {
                 fetchData :function(){
                    $.ajax({
                       dataType:'json',
                       url:'/j/users',
                       method:'GET'
                    })
                    .done(result=>{
                       this.users = result.users;
                       $(el).show('slow');
                    })
                    .fail(()=>console.log('Error!'))
                    .always(()=>console.log('Fin!'));
                 }
             }
        });
      });
    });
  </script>
```

А здесь модель вытягивает данные методом **jQuery**

http://kodaktor.ru/handlebars_step3

```
<!DOCTYPE html>
<html>
 <head>
  <title>Handlebars3</title><meta charset="utf-8"><link rel="stylesheet" href="/css/board3">
  <script src="//cdnjs.cloudflare.com/ajax/libs/handlebars.js/1.3.0/handlebars.min.js"></script>
  <script src="//ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>
 </head>
 <body>

    <h1>Handlebars3 <a target="_blank" href="http://codepen.io/gossoudarev/pen/XNWKVO?editors=1111">codepen</a></
    <button id='b'> click me </button>

    <div id='r'></div>
    <script id='userslist' type='text/x-handlebars-template'>
     <ul>
       {{#each users}}
         <li>
            {{login}} :  {{password}}
         </li>
       {{/each}}
     </ul>
   </script>

   <script>
     $(()=>{
        $('#b').on('click', ()=>{
            let render = (view,context) => Handlebars.compile( $('#'+view).html())(context) ;
            $.ajax({
                url: 'http://kodaktor.ru/json/users', method: 'GET'})
             .done(result=>{
                 $('#r').html (  render('userslist',{"users":result.users})  );
             })
             .fail(e=>alert(JSON.stringify(e)));
        });
     });
   </script>
 </body>
</html>
```

HandleBars работает
и на клиенте и на сервере
ср. синтаксис с **Vue**


```
ReactDOM.render((
        <Router>
                <Route path="/" component={MainLayout}>
                        <IndexRoute component={Home} />
                        <Route component={SearchLayout}>
                                <Route path="users" component={UserList} />
                                <Route path="widgets" component={WidgetList} />
                        </Route>
                </Route>
        </Router>
), document.getElementById('root'));
```

А как дела обстоят с **XML?**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<?xml-stylesheet type="text/xsl" href="users1.xsl"?>
<users xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="users.xsd" >

    <user login="student" password="tneduts" />
    <user login="myuser" password="mypas" />
    <user login="teacher" password="qq" />
    <user login="myking" password="myqueen" />
</users>
```

```
<?xml version="1.0" encoding="utf-8" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

 <xsl:template match="/">
    <html><head><title></title></head>
    <body>
        <ul>
            <xsl:for-each select="/users/user">
               <li> <xsl:value-of select="@login"/> </li>
            </xsl:for-each>
        </ul>
    </body></html>
 </xsl:template>
</xsl:stylesheet>
```

На что это больше похоже:

HandleBars или **Vue?**

```xml
<?xml version="1.0" encoding="utf-8" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

 <xsl:template match="/users">
    <html><head><title></title></head>
    <body>
        <ul>
            <xsl:apply-templates/>
        </ul>
    </body></html>
 </xsl:template>

 <xsl:template match="user">
    <li> <xsl:value-of select="@login"/> </li>
 </xsl:template>

</xsl:stylesheet>
```

Шаблон / гнездо для **user**-ов выносится за пределы родителя

```xml
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:simpleType name="txt">
        <xs:restriction base="xs:string">
        <xs:maxLength value="32" />
        </xs:restriction>
    </xs:simpleType>

    <xs:attribute name="login" type="txt"  />
    <xs:attribute name="password" type="txt"  />

    <xs:complexType name="usr">
       <xs:attribute ref="login" use="required" />
       <xs:attribute ref="password" use="required" />
    </xs:complexType>

    <xs:element name="user"  type="usr" />

    <xs:complexType name="tabl">
        <xs:sequence>
          <xs:element ref="user"  minOccurs="1" maxOccurs="unbounded"   />
        </xs:sequence>
    </xs:complexType>

    <xs:element name="users" type="tabl" />

</xs:schema>
```

А это схема