

PeerGaming



Share the Fun



Info



Web & Game
Engineer

@Autarc = [, ]

Overview

Idea

Design

Code

Background

Motivation

people interact

Motivation

people interact



play together

Problem

gamers - limited selection

Problem

gamers - limited selection

devs - complex setup

#noBackend / #yoFrontend

Solution

Creating a Client-Side Multiplayer Gaming Framework for the Web, which handles Distributed Logic on Multiple Systems in "Real-Time".

Foundation

Browser Based Games

DOM - Canvas - WebGL

Browser Based Games

DOM - Canvas - WebGL

Mobile Market

Multiplayer

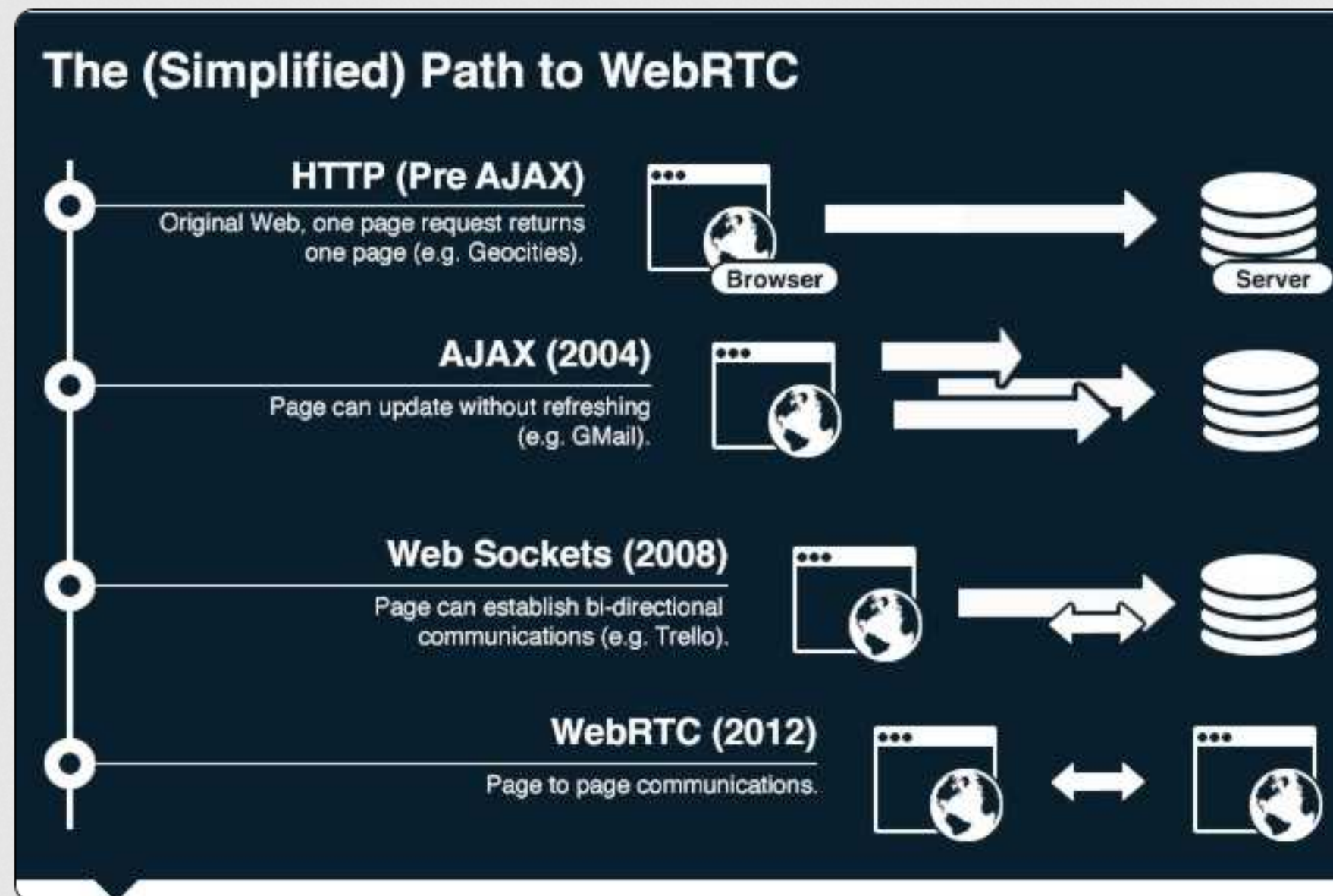
WebSockets or XHR-Polling

Multiplayer

WebSockets or XHR-Polling

Client-Server Architecture

Evolution



Graphic by Jimmy Lee / jimmylee.info

WebRTC

Technology

WebRTC



PeerConnection

WebRTC



PeerConnection



MediaStream

WebRTC



PeerConnection



MediaStream



DataChannel

Structure

Topology

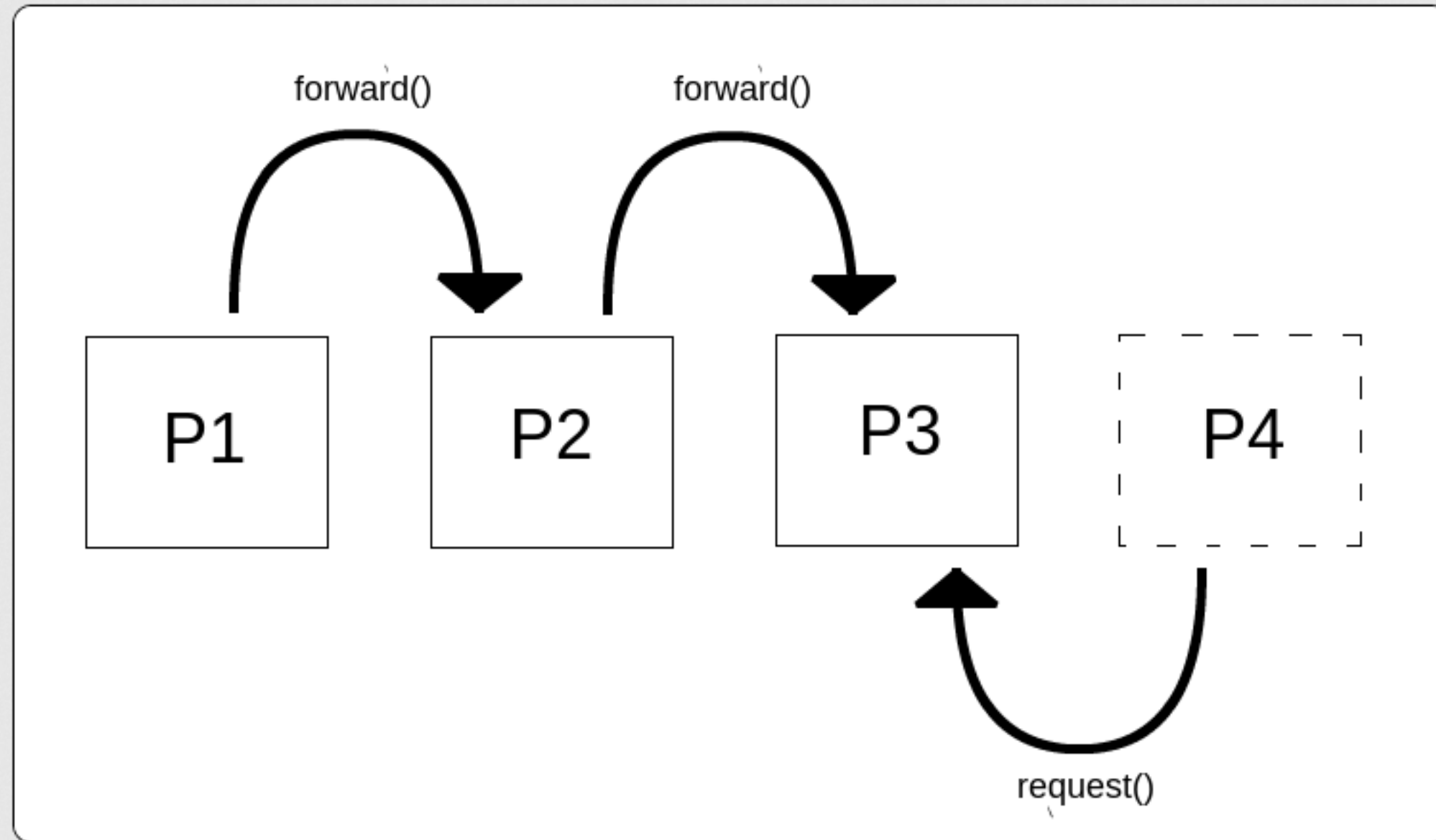
Subnetting

Topology

Subnetting

Full Mesh $\Rightarrow 1 : N$

Bootstrap



Implementation

Components

```
/** API Overview */  
  
var pg = {  
  
  noConflict    : fn - 'reset namespace'  
  VERSION      : obj - 'refers to the current version'  
  info         : obj - 'information about the state'  
  
  // config    : fn - 'configuration for the network'  
  // login     : fn - 'set identifier and create player'  
  // player    : obj - 'own user instance (writeable)'  
  // peers     : obj - 'list of connected players (readable)'  
  // data      : arr - 'shortcut to access peers.data + player.data'  
  // sync      : obj - 'synchronized shared object'  
  // loop      : fn - 'synchronized rendering process'  
  // channel   : fn - 'handler for a "Channel".'  
  // game      : fn - 'handler for a "Game".'  
  // routes    : fn - 'define default and custom routes'  
};
```

Components

```
/** API Overview */  
  
var pg = {  
  
  // noConflict : fn - 'reset namespace'  
  // VERSION    : obj - 'refers to the current version'  
  // info       : obj - 'information about the state'  
  
  config       : fn - 'configuration for the network'  
  login        : fn - 'set identifier and create player'  
  
  // player     : obj - 'own user instance (writeable)'  
  // peers      : obj - 'list of connected players (readable)'  
  // data       : arr - 'shortcut to access peers.data + player.data'  
  // sync       : obj - 'synchronized shared object'  
  // loop       : fn - 'synchronized rendering process'  
  // channel    : fn - 'handler for a "Channel".'  
  // game       : fn - 'handler for a "Game".'  
  // routes     : fn - 'define default and custom routes'  
};
```

Components

```
/** API Overview */  
  
var pg = {  
  
  // noConflict : fn - 'reset namespace'  
  // VERSION    : obj - 'refers to the current version'  
  // info       : obj - 'information about the state'  
  // config     : fn - 'configuration for the network'  
  // login      : fn - 'set identifier and create player'  
  
  player       : obj - 'own user instance (writeable)'  
  peers        : obj - 'list of connected players (readable)'  
  data         : arr - 'shortcut to access peers.data + player.data'  
  sync         : obj - 'synchronized shared object'  
  loop         : fn  - 'synchronized rendering process'  
  
  // channel    : fn - 'handler for a "Channel".'  
  // game       : fn - 'handler for a "Game".'  
  // routes     : fn - 'define default and custom routes'  
};
```

Components

```
/** API Overview */
```

```
var pg = {
```

```
  // noConflict : fn - 'reset namespace'  
  // VERSION    : obj - 'refers to the current version'  
  // info       : obj - 'information about the state'  
  // config     : fn - 'configuration for the network'  
  // login      : fn - 'set identifier and create player'  
  // player     : obj - 'own user instance (writeable)'  
  // peers      : obj - 'list of connected players (readable)'  
  // data       : arr - 'shortcut to access peers.data + player.data'  
  // sync       : obj - 'synchronized shared object'  
  // loop       : fn - 'synchronized rendering process'
```

```
  channel      : fn - 'handler for a "Channel"'  
  game         : fn - 'handler for a "Game"'  
  routes       : fn - 'define default and custom routes'
```

```
};
```

Account

```
/** define a name through user input */  
  
// with plain text  
pg.login( 'Autarc' );  
  
// later with a 3rd party service via OAuth  
pg.login( 'Autarc', 'Github' );
```


Rooms = Channel / Game

```
/**
 * default scheme:
 *
 *   channel - 1 parameter || http://{HOST}.{TLD}/#!/{channel}/
 *   game    - 2 parameter || http://{HOST}.{TLD}/#!/{channel}/{id}/
 */

// example for a game
http://peergaming.net/pg-cycle/42

// handle game events
pg.game( 'pg-cycle', function ( game ) { ... } );
```


Peerlist (Server)

```
/** select a random peer for the initial connection */  
  
// addr: 'pg-cycle/42',  
// id   : 'a1097elf-4d28-4695-b7ac-0e399690040e'  
  
function getPartner ( addr, id ) {  
  
    var keys = Object.keys( rooms[ addr ] ),  
        partnerID = null;  
  
    if ( keys.length > 1 ) partnerID = keys[~~(Math.random() * keys.length)];  
    return ( partnerID !== id ) ? partnerID : this.getPartner( addr, id );  
}
```

Network

PeerRouting (P2P)

```
/** use brokering for new connections **/  
  
function send ( action, data ) {  
  
    var remote = this.info.remote;  
  
    // use an already known peer  
    if ( this.info.transport ) {  
        var proxy = { action: action, local: instance.id, remote: remote };  
        return this.info.transport.send( 'register', data, proxy );  
    }  
  
    // send via server  
    socket.send({ action: action, data: data, remote: remote });  
}
```

Network

***** manually *****

```
/** alternative hook for custom handling the credential exchange */  
  
pg.config.noServer( function( msg, conn ) {  
  
    // show own information ( iceCandidates, SDP packages )  
    document.body.innerHTML += msg.type + ':' + msg.data;  
  
    // reference to add a partners configurations  
    function setOwnCredentials( msg ) {  
        conn.set( msg.type, msg.data );  
    }  
  
});
```

Reactive Data

```
/** using getter & setter for values */  
  
// define internal reference  
var player = pg.player.data;  
  
player.posX = 1;  
player.keys = [ 'a', 'b', 'c', 'd' ];  
  
// shared object between all peers  
pg.sync.state = 'start';  
  
pg.sync.cookies = [];  
pg.sync.cookies.push({ x: 100, y: 50, r: 20 });
```

Synchronized Rendering

```
/** automatic synchronisation */  
  
// define internal reference  
var entries = pg.data;  
  
pg.loop( function render ( dt ) {  
  
    for ( var i = 0, l = entries.length; i < l; i++ ) {  
  
        console.log( entries[i] );  
    }  
  
});
```

Example

Quick Start

1.) include script tag

Quick Start

- 1.) include script tag
- 2.) setup room handler

Quick Start

- 1.) include script tag
- 2.) setup room handler
- 3.) login - create user

Quick Start

- 1.) include script tag
- 2.) setup room handler
- 3.) login - create user
- 4.) initialize the game

Quick Start

- 1.) include script tag
- 2.) setup room handler
- 3.) login - create user
- 4.) initialize the game
- 5.) use player, data & sync

Tip

Hint: Focus on Input

pg-catch

<http://demo.peergaming.net>

Conclusion

Features

standalone

Features

standalone

configurable

Features

standalone

configurable

simple

Plans

room options

Plans

room options

audio-streams

support via form

PeerGaming - Statistics

I would appreciate to get some feedback about the usage and demand of browser based games. Hopefully this can provide some insight about your demand - which allows me to improve the framework or even help others on the current market.

Thanks for taking a few minutes and filling out the survey!

1.) Which role describes you best in a project ?

- ☐ Developer
- ☐ Designer
- ☐ Manager
- ☐ Tester

2.) Did you made / tried to create a browser based game before ?

- ☐ Yes
- ☐ No

2.*) If yes, did this game include a multiplayer aspect ?
(just skip if you didn't worked on a game before)

- ☐ Yes
- ☐ No

3.) What kind of network design (architecture) do you prefer ?

- ☐ Server-Client
- ☐ Peer-to-Peer

4.) Which kind of genre do you prefer in games ?
(just skip if you don't play any kind of game)

- ☐ Strategy
- ☐ Jump & Run
- ☐ Roleplaying
- ☐ Adventure
- ☐ Action
- ☐ Shooter
- ☐ Sport

Links

peer gaming.net

```
bower install peer gaming
```

Questions & Discussion