Social Games

und ihr Clean Code

EA
SPORTS
FIFA 13
FIFA

12.79 Mio.

http://www.vgchartz.com/gamedb/?name=fifa+soccer+13 || http://cartelthemes.com/fifa-13-193-hd-wallpapers.html

CALL OF DUTY
MW3

30.11 Mio.

http://www.vgchartz.com/gamedb/?name=Call+of+Duty%3A+Modern+Warfare+3 || http://www.alextv.de/2011/11/10/call-of-duty-modern-warfare-3-review

Video Games = spezielle Interessen

535.710
DIAMOND DASH
16

159.79 Mio.

Social Games = Spiele für Jeden

Berlin

2009 — 20 Angestellte

Heute — 250 Mitarbeiter

Magic Land

Pocket Village

Pearl's Peril

Bubble Island

20 Game-Teams

Fantastic Forest

Kingsbridge

Diamond Dash

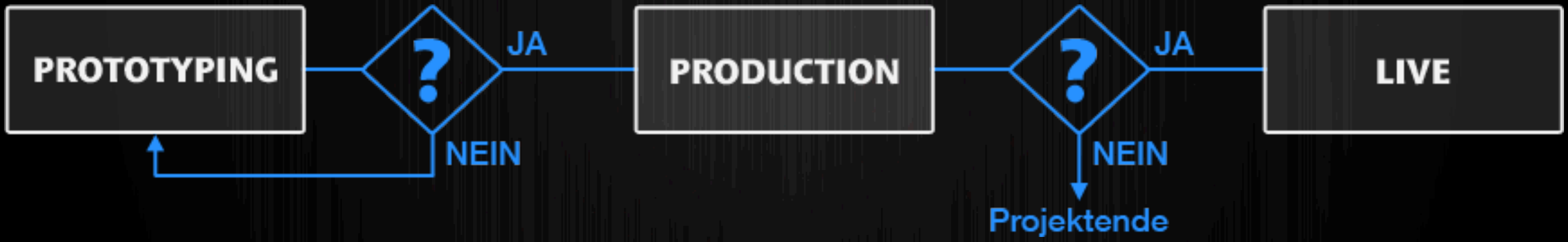
Monster World

Unabhängige Entwicklungsteams

Veränderliche, kleine Teams

Einheitlichkeit

Arbeitsweise



PROTOTYPING

?

JA

NEIN

PRODUCTION

?

JA

NEIN

Projektende

LIVE

Keine festen Rollen

Hohe Verantwortung

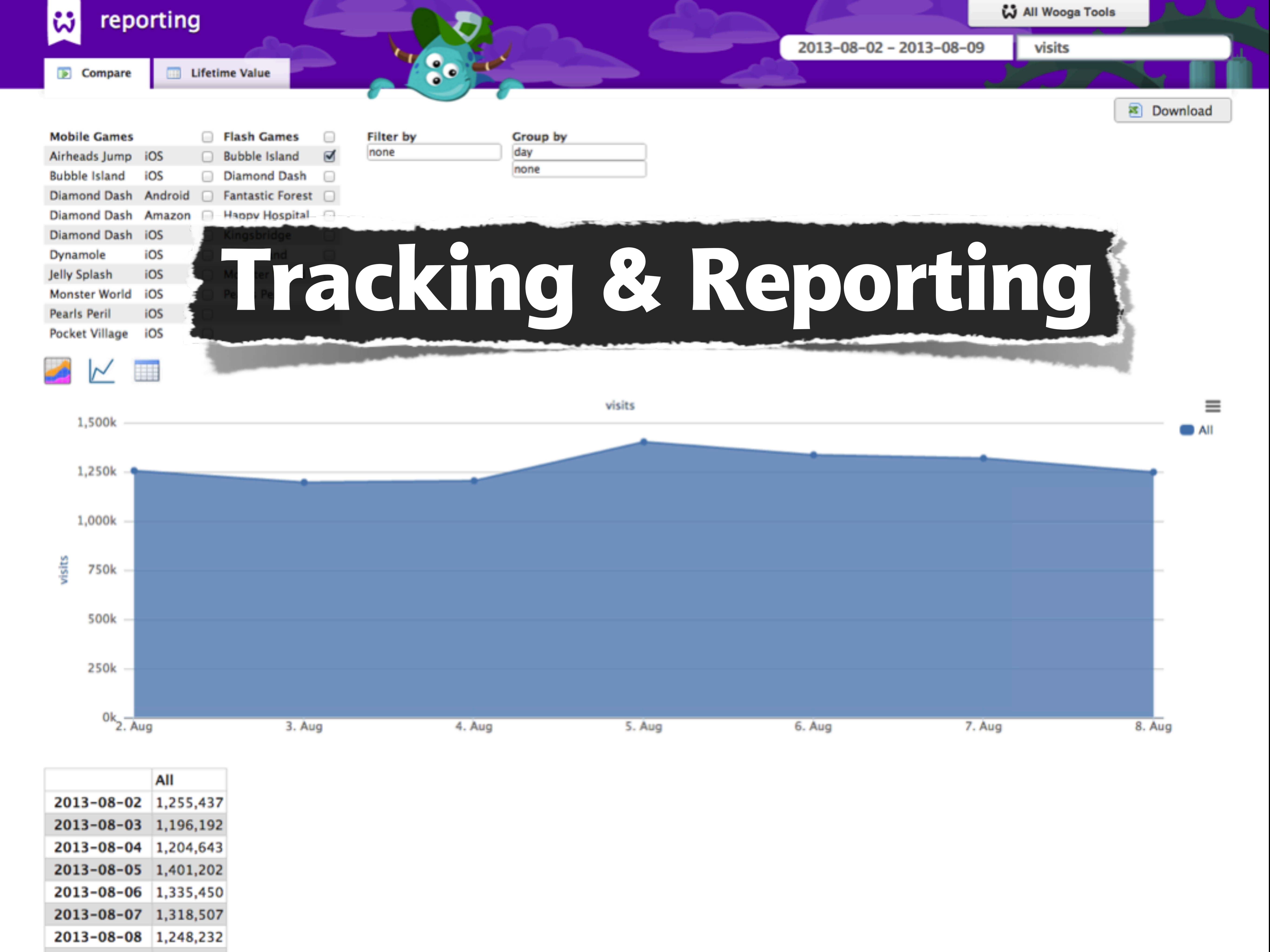
Konfigurierbarkeit

Wartbarkeit

MAI
JULI
AUGUST
JUNI
SEPTEMBER

Wöchentliche Releases

Tracking & Reporting


reporting
Compare
Lifetime Value
All Wooga Tools
Download
visits
1,500k
1,250k
1,000k
750k
500k
250k
0k
2. Aug
3. Aug
4. Aug
5. Aug
6. Aug
7. Aug
8. Aug
All

2013-08-02 – 2013-08-09

visits

Mobile Games		☐	Flash Games	☐
Airheads Jump	iOS	☐	Bubble Island	☑
Bubble Island	iOS	☐	Diamond Dash	☐
Diamond Dash	Android	☐	Fantastic Forest	☐
Diamond Dash	Amazon	☐	Happy Hospital	☐
Diamond Dash	iOS		Kingsbridge	
Dynamole	iOS			
Jelly Splash	iOS			
Monster World	iOS			
Pearls Peril	iOS			
Pocket Village	iOS			

Filter by: none

Group by: day, none

	All
2013-08-02	1,255,437
2013-08-03	1,196,192
2013-08-04	1,204,643
2013-08-05	1,401,202
2013-08-06	1,335,450
2013-08-07	1,318,507
2013-08-08	1,248,232

Agile Arbeitsweise

Agilität

Einheitlichkeit

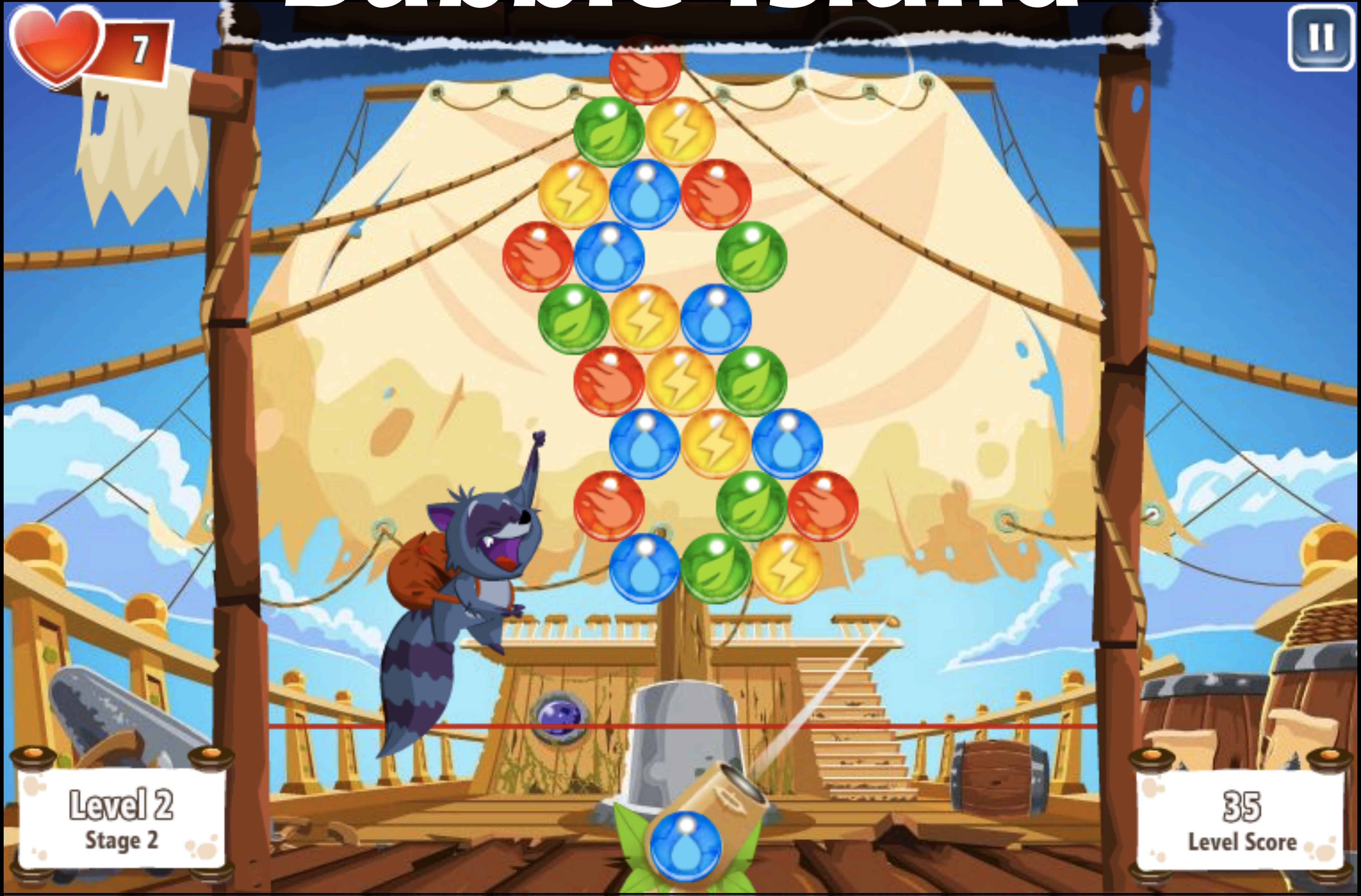
↓ Verbessert

Wartbarkeit

↓ Ermöglicht

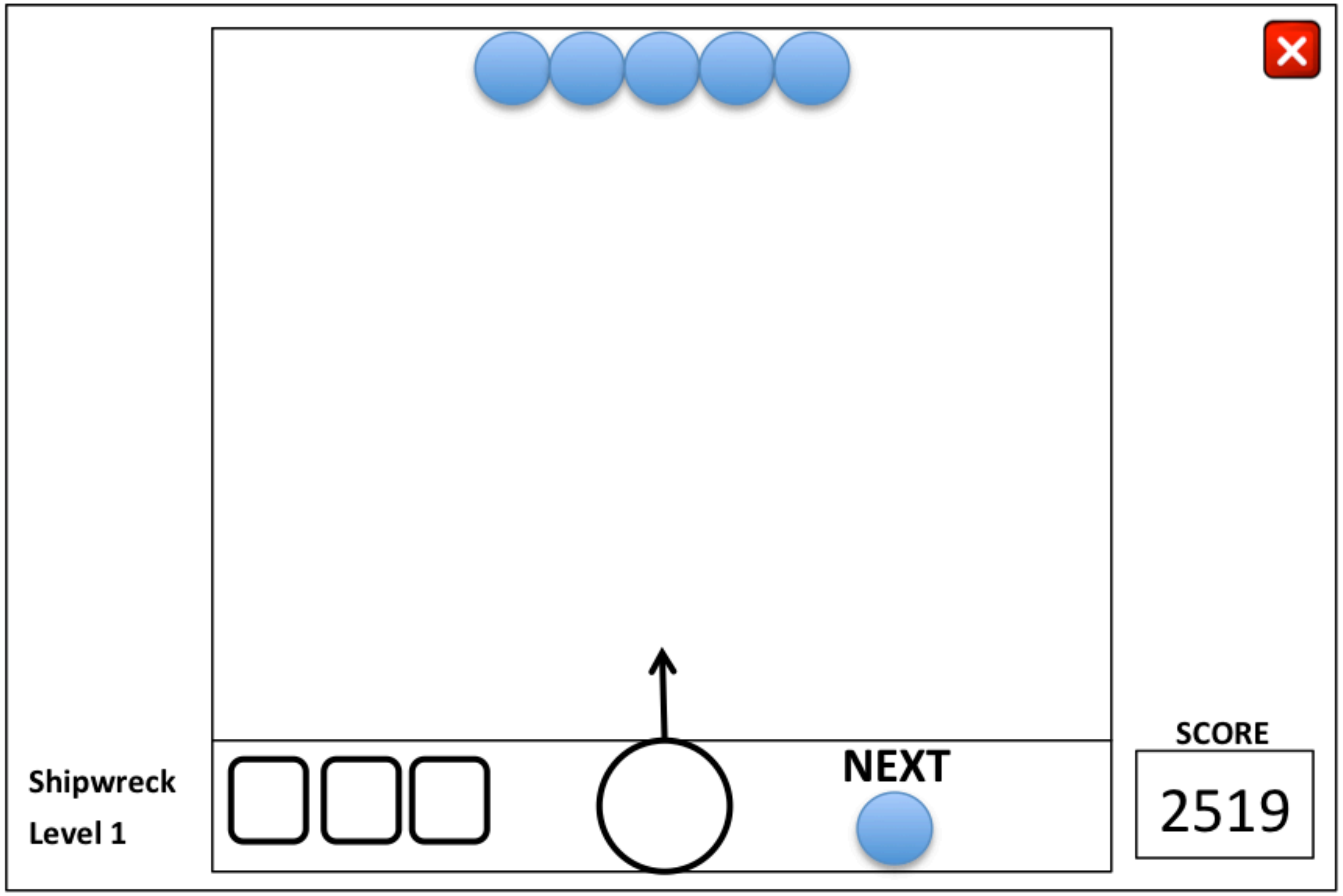
Agilität

Bubble Island


7
Level 2
Stage 2
35
Level Score

2009

Entwicklungsstart


Shipwreck
Level 1
NEXT
SCORE
2519

2011

Nicht mehr wartbar!

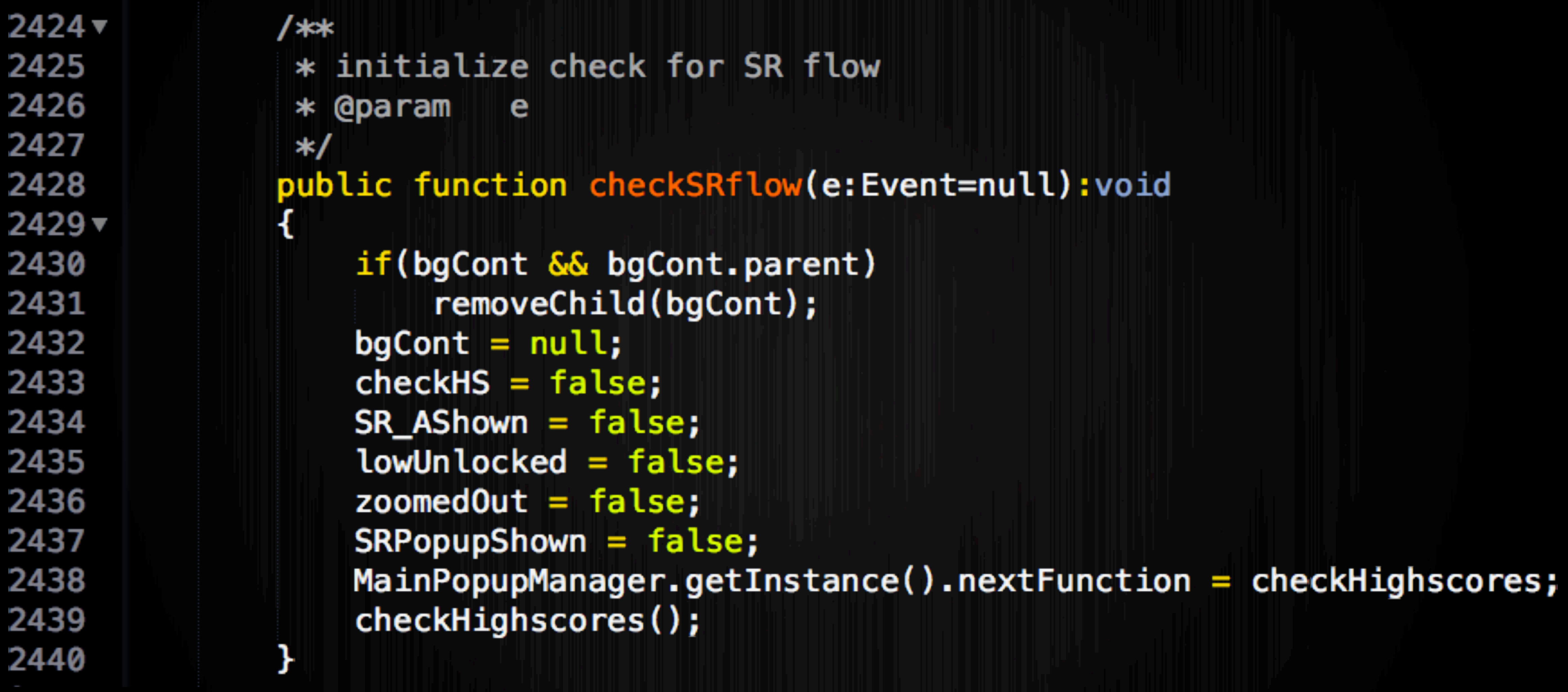
http://www.flickr.com/photos/mecklenburg

Aber immer noch erfolgreich!

Refactoring!

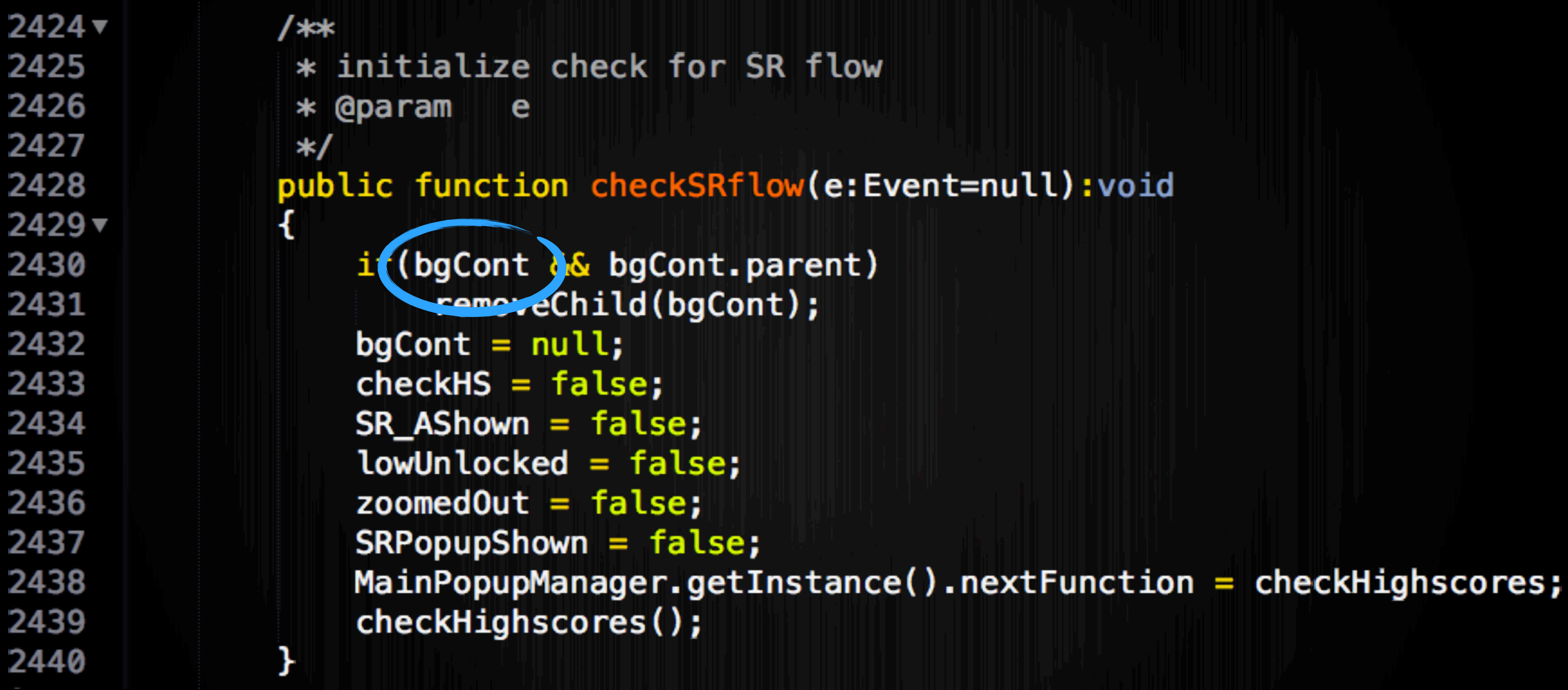
1

Einheitlichkeit



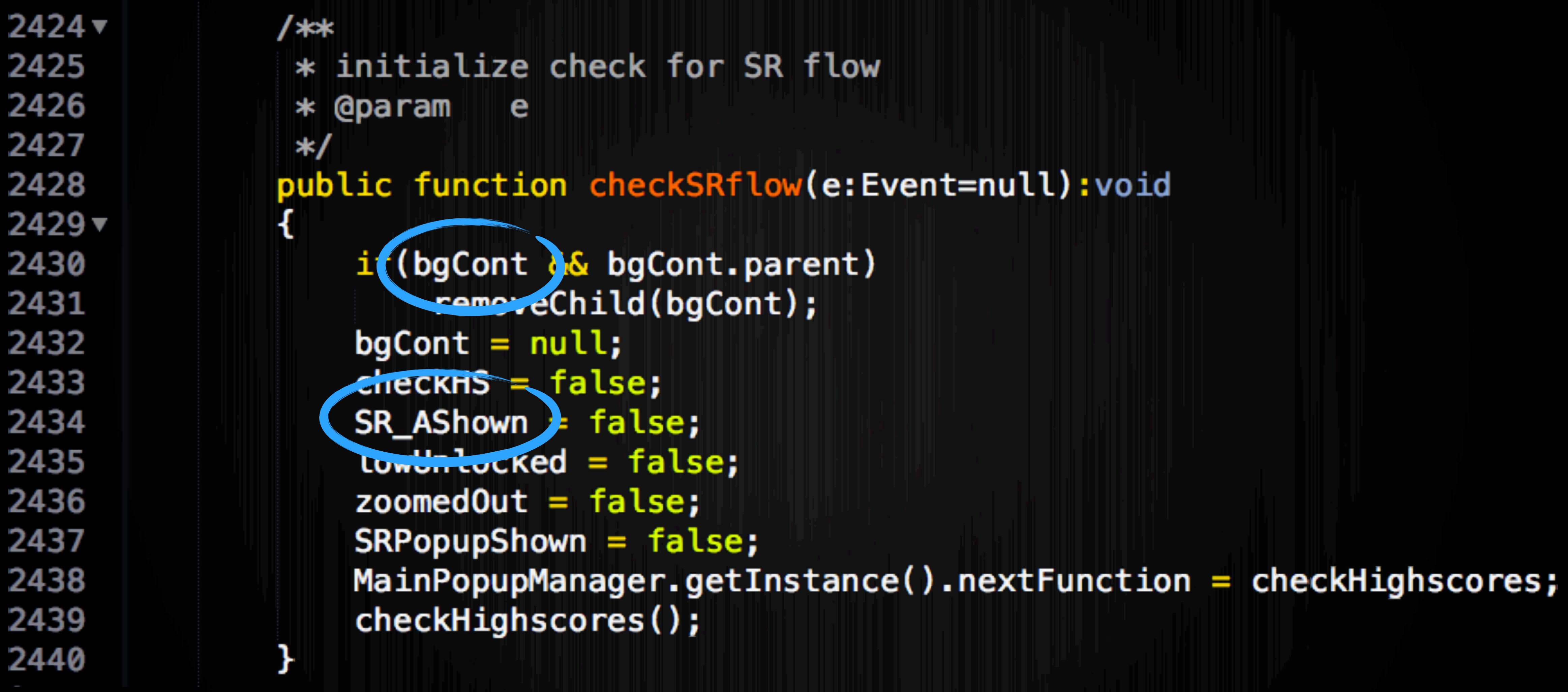
```
/**
 * initialize check for SR flow
 * @param   e
 */
public function checkSRflow(e:Event=null):void
{
    if(bgCont && bgCont.parent)
        removeChild(bgCont);
    bgCont = null;
    checkHS = false;
    SR_AShown = false;
    lowUnlocked = false;
    zoomedOut = false;
    SRPopupShown = false;
    MainPopupManager.getInstance().nextFunction = checkHighscores;
    checkHighscores();
}
```

Einheitlichkeit



```
/**
 * initialize check for SR flow
 * @param   e
 */
public function checkSRflow(e:Event=null):void
{
    if(bgCont && bgCont.parent)
        removeChild(bgCont);
    bgCont = null;
    checkHS = false;
    SR_AShown = false;
    lowUnlocked = false;
    zoomedOut = false;
    SRPopupShown = false;
    MainPopupManager.getInstance().nextFunction = checkHighscores;
    checkHighscores();
}
```

Einheitlichkeit



```
/**
 * initialize check for SR flow
 * @param   e
 */
public function checkSRflow(e:Event=null):void
{
    if(bgCont && bgCont.parent)
        removeChild(bgCont);
    bgCont = null;
    checkHS = false;
    SR_AShown = false;
    townUnlocked = false;
    zoomedOut = false;
    SRPopupShown = false;
    MainPopupManager.getInstance().nextFunction = checkHighscores;
    checkHighscores();
}
```

Einheitlichkeit

Namensgebung

Namensgebung

- Sprechende Namen

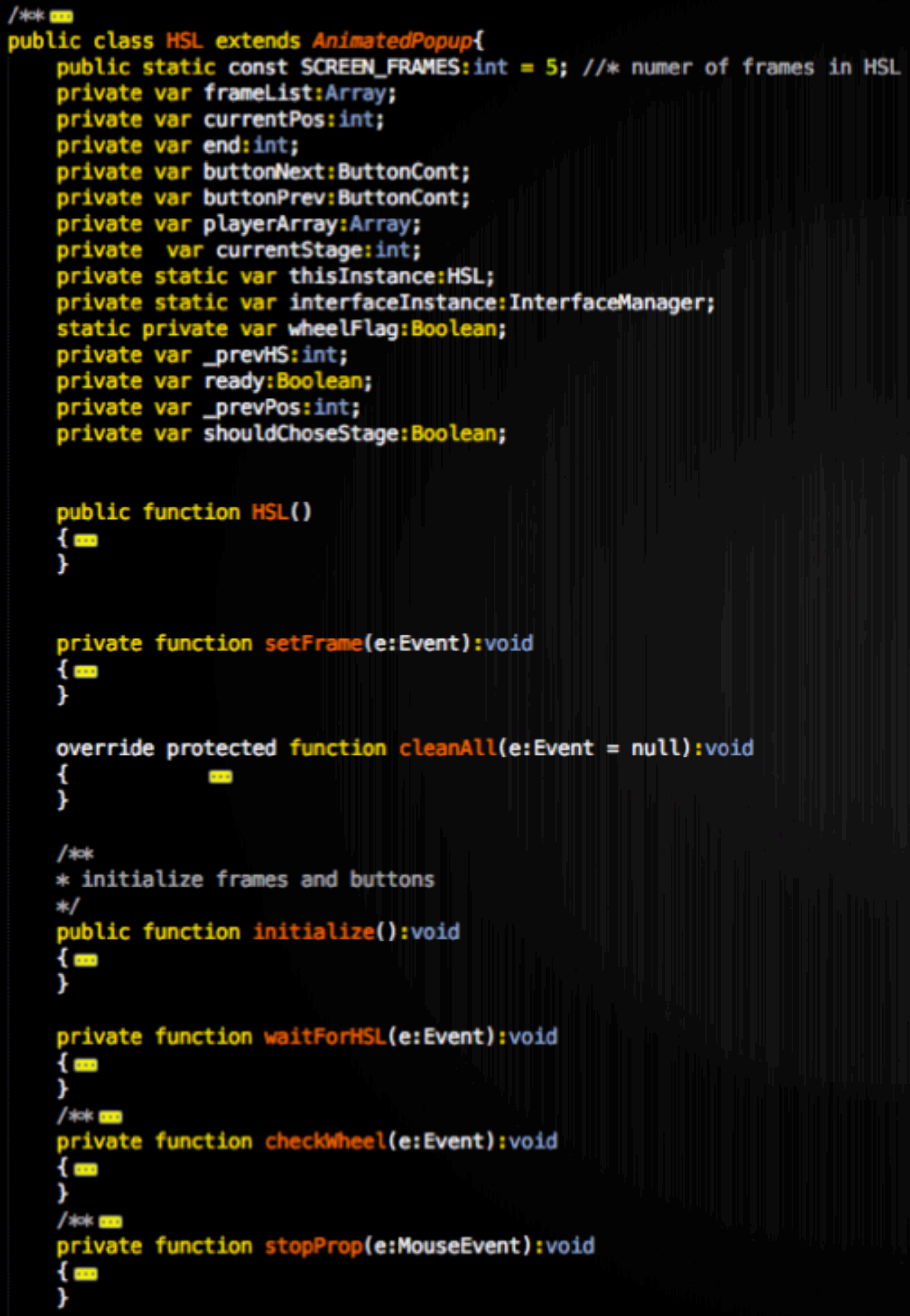
Namensgebung

- Sprechende Namen
- Keine Abkürzungen

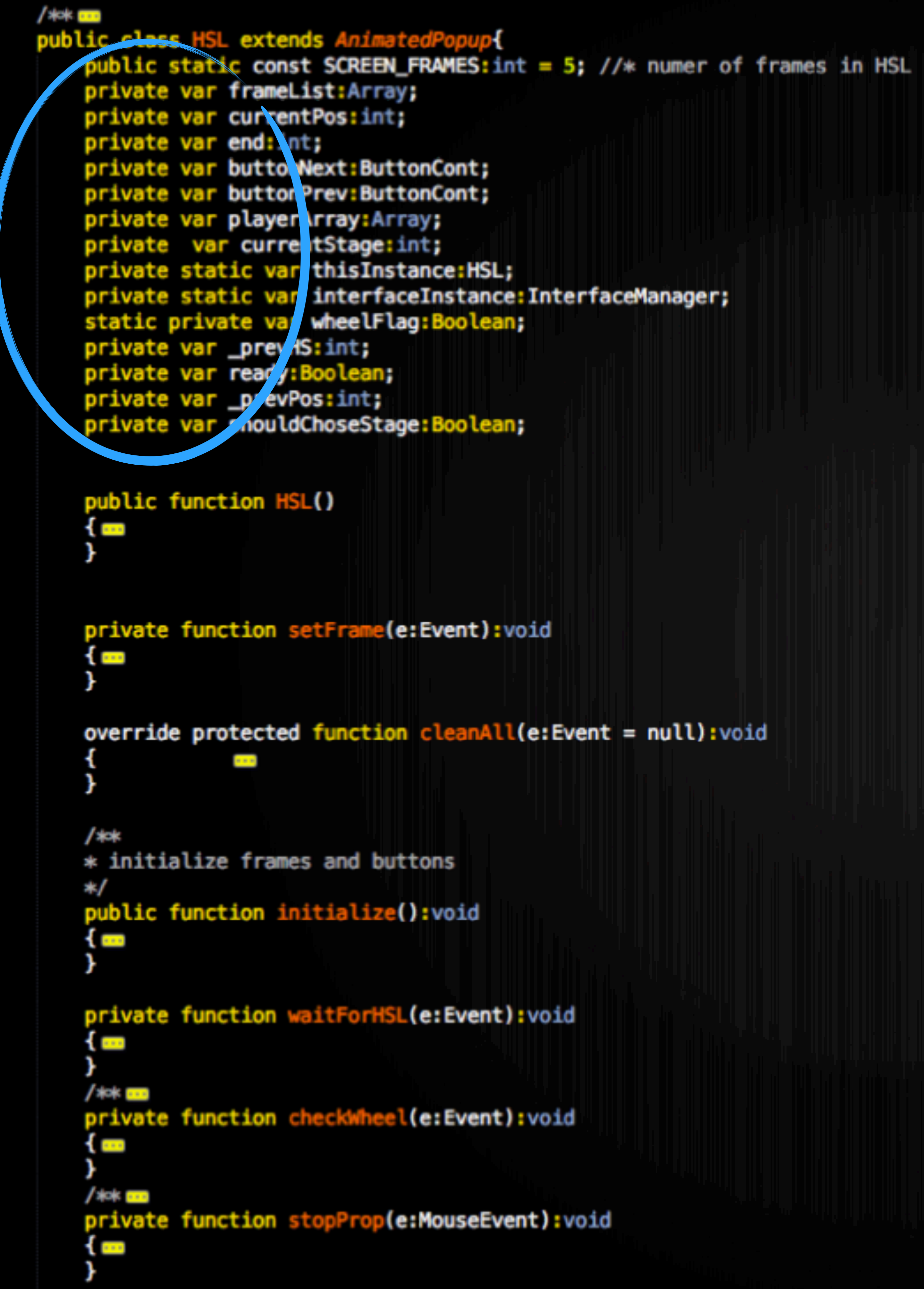
Namensgebung

- Sprechende Namen
- Keine Abkürzungen
- 1 Begriff pro Konzept

Einheitlichkeit

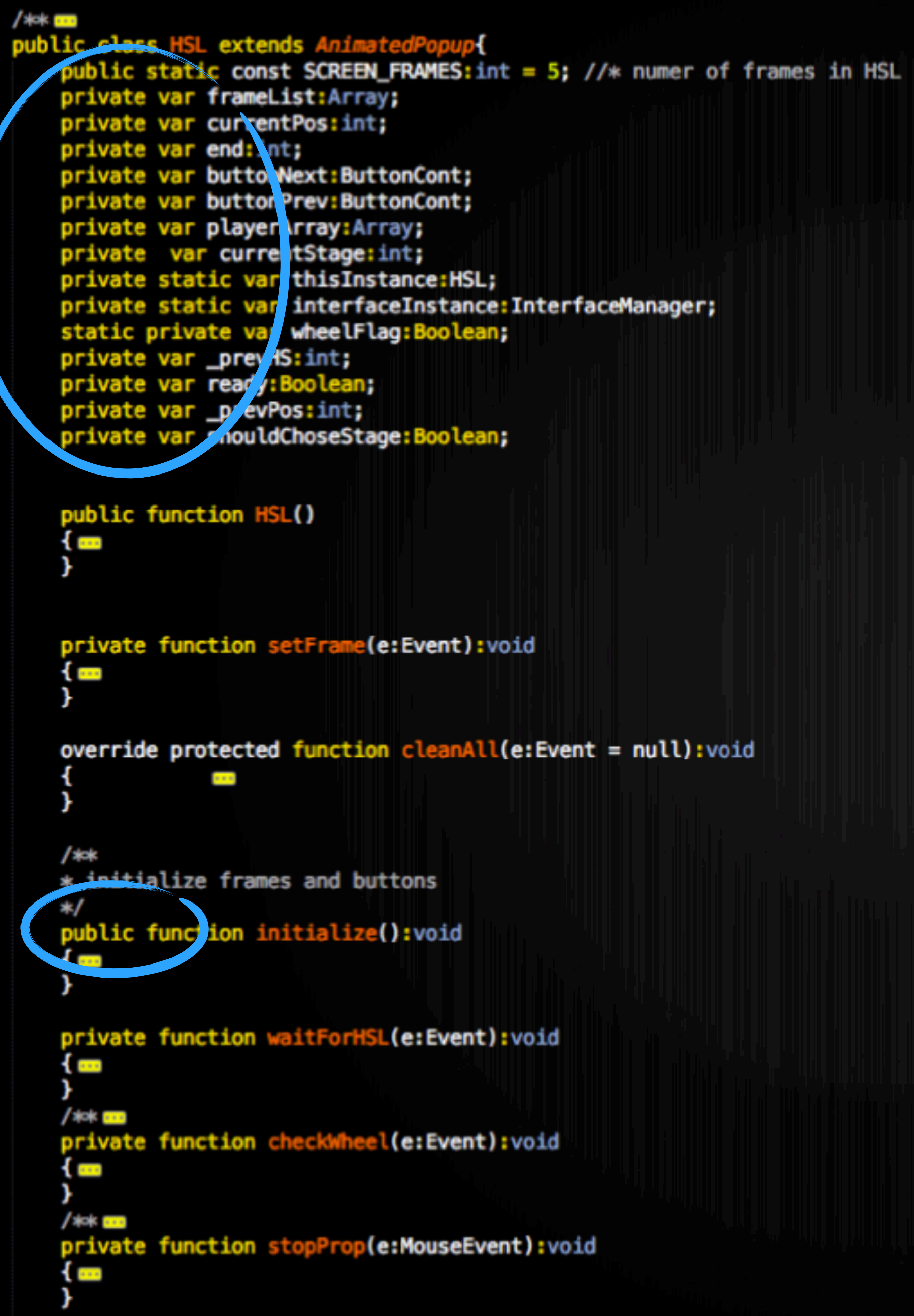


```
/** …
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;


    public function HSL()
    {…
    }


    private function setFrame(e:Event):void
    {…
    }

    override protected function cleanAll(e:Event = null):void
    {           …
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {…
    }

    private function waitForHSL(e:Event):void
    {…
    }
    /** …
    private function checkWheel(e:Event):void
    {…
    }
    /** …
    private function stopProp(e:MouseEvent):void
    {…
    }
```



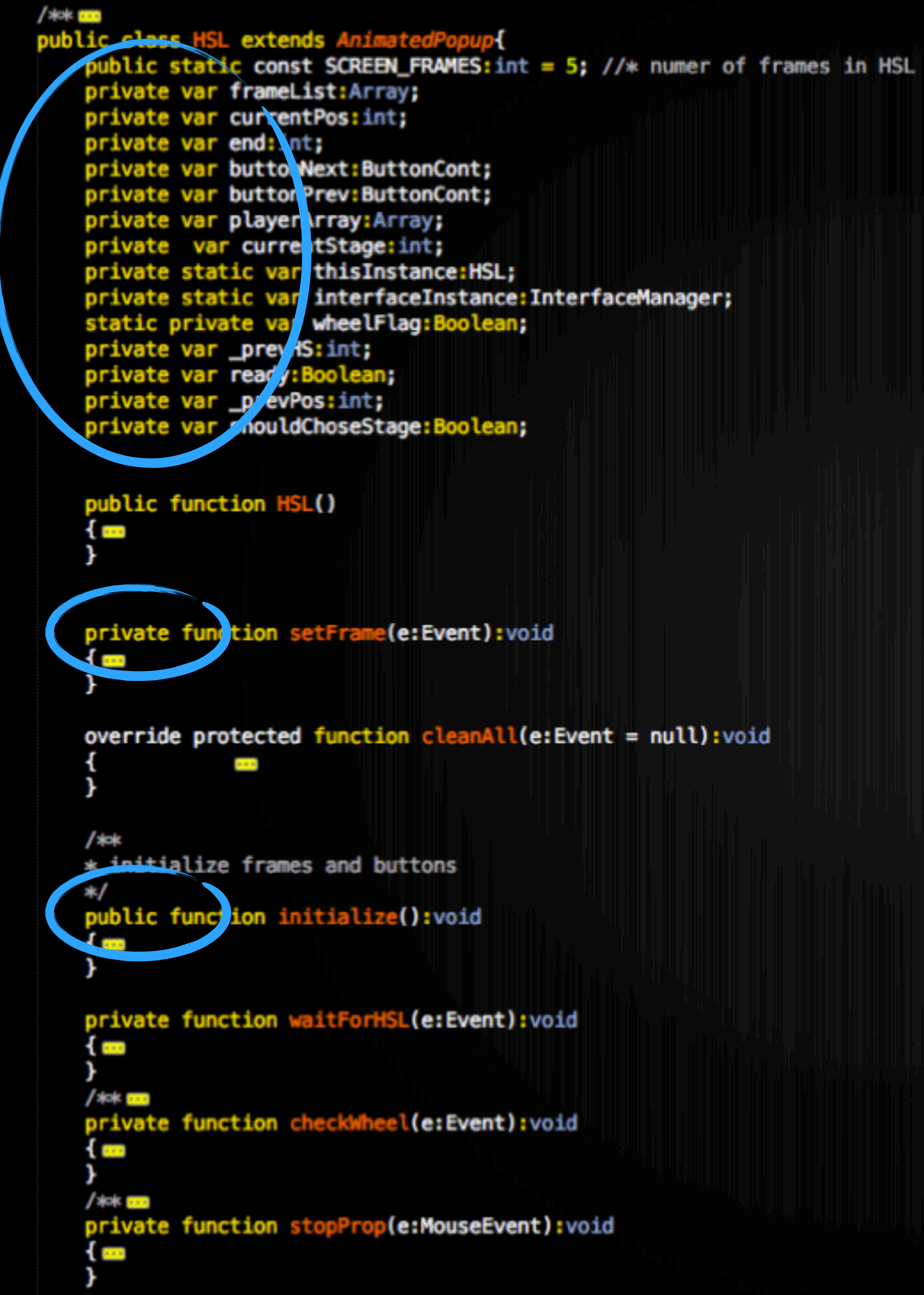
```
/** ...
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;


    public function HSL()
    {...
    }


    private function setFrame(e:Event):void
    {...
    }

    override protected function cleanAll(e:Event = null):void
    {           ...
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {...
    }

    private function waitForHSL(e:Event):void
    {...
    }
    /** ...
    private function checkWheel(e:Event):void
    {...
    }
    /** ...
    private function stopProp(e:MouseEvent):void
    {...
    }
```

Einheitlichkeit



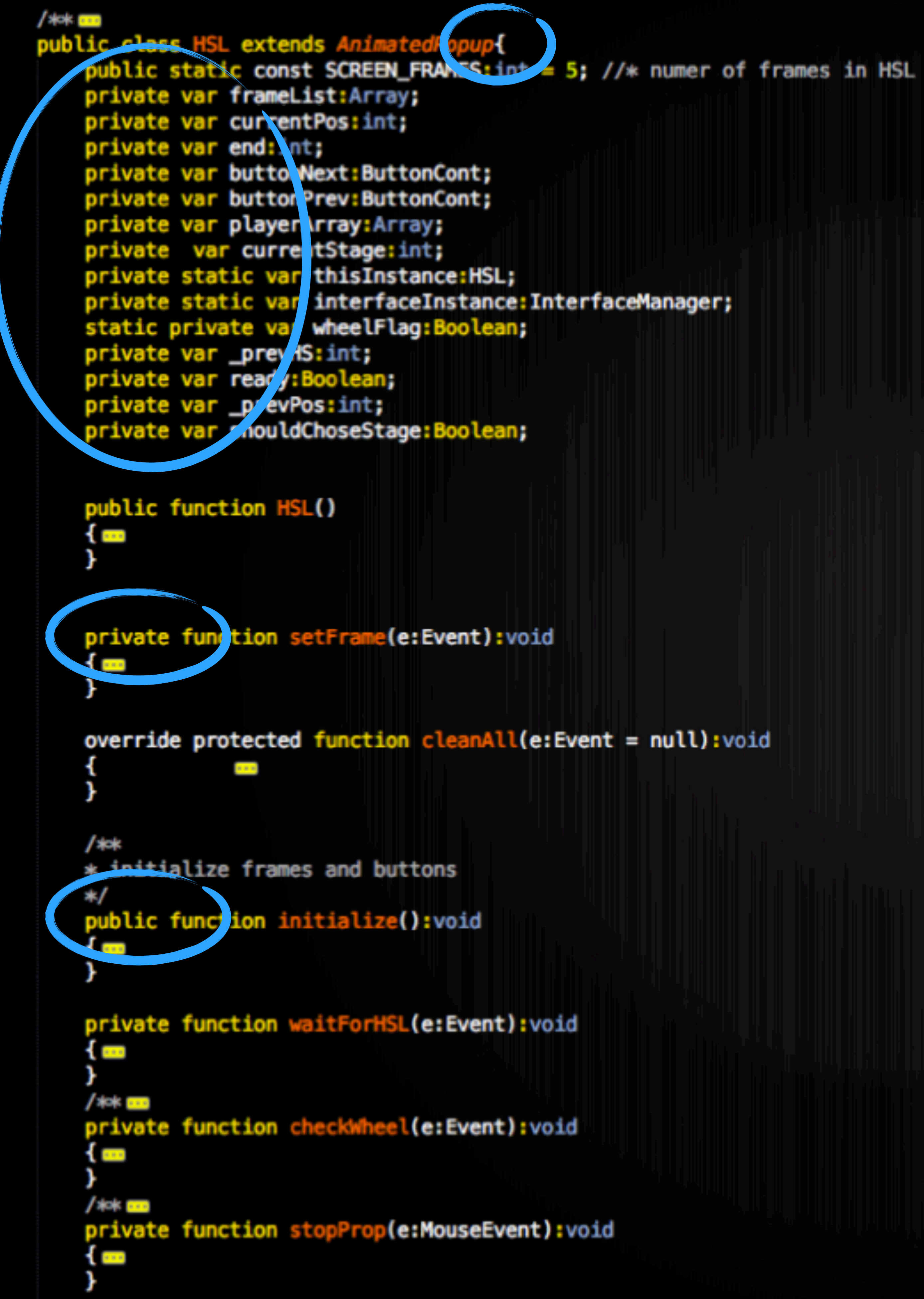
```
/** ...
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;


    public function HSL()
    {...
    }


    private function setFrame(e:Event):void
    {...
    }

    override protected function cleanAll(e:Event = null):void
    {           ...
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {...
    }

    private function waitForHSL(e:Event):void
    {...
    }
    /** ...
    private function checkWheel(e:Event):void
    {...
    }
    /** ...
    private function stopProp(e:MouseEvent):void
    {...
    }
```

Einheitlichkeit



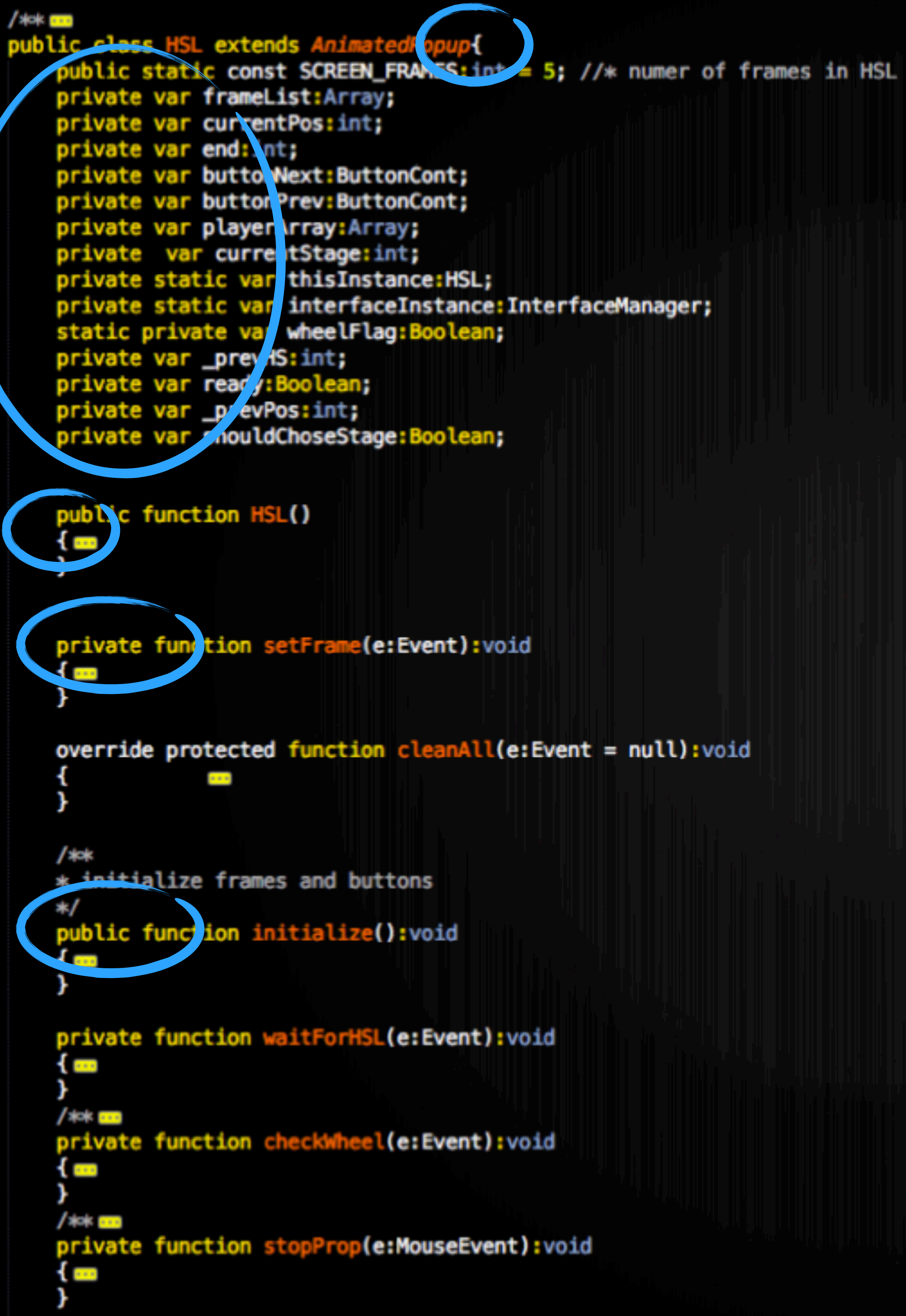
```
/** …
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;


    public function HSL()
    {…
    }


    private function setFrame(e:Event):void
    {…
    }

    override protected function cleanAll(e:Event = null):void
    {            …
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {…
    }

    private function waitForHSL(e:Event):void
    {…
    }
    /** …
    private function checkWheel(e:Event):void
    {…
    }
    /** …
    private function stopProp(e:MouseEvent):void
    {…
    }
```

Einheitlichkeit



```
/**
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;


    public function HSL()
    {
    }


    private function setFrame(e:Event):void
    {
    }

    override protected function cleanAll(e:Event = null):void
    {
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {
    }

    private function waitForHSL(e:Event):void
    {
    }
    /**
    private function checkWheel(e:Event):void
    {
    }
    /**
    private function stopProp(e:MouseEvent):void
    {
    }
```

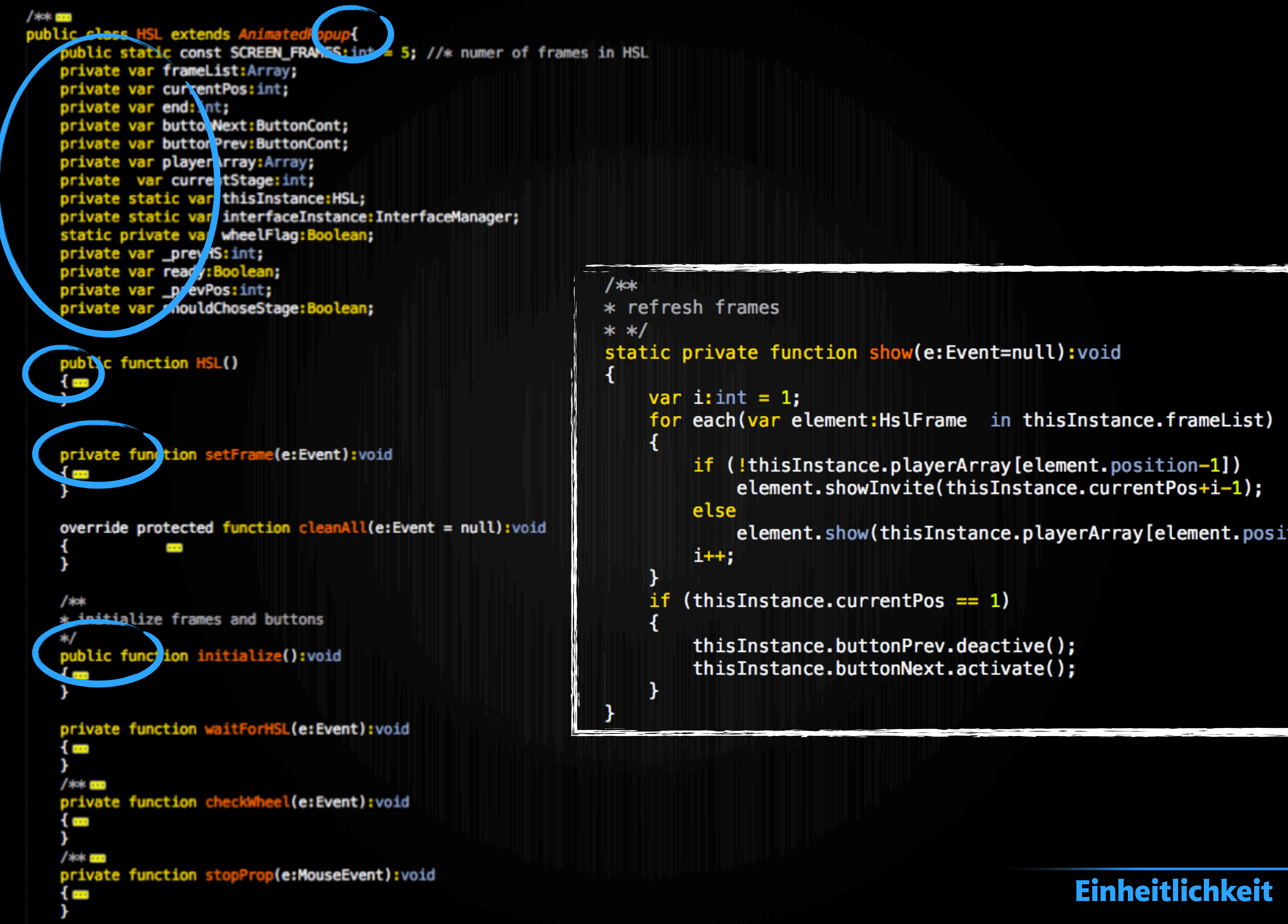
Einheitlichkeit



```
/** ...
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;

    public function HSL()
    {...
    }

    private function setFrame(e:Event):void
    {...
    }

    override protected function cleanAll(e:Event = null):void
    {           ...
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {...
    }

    private function waitForHSL(e:Event):void
    {...
    }
    /** ...
    private function checkWheel(e:Event):void
    {...
    }
    /** ...
    private function stopProp(e:MouseEvent):void
    {...
    }
```

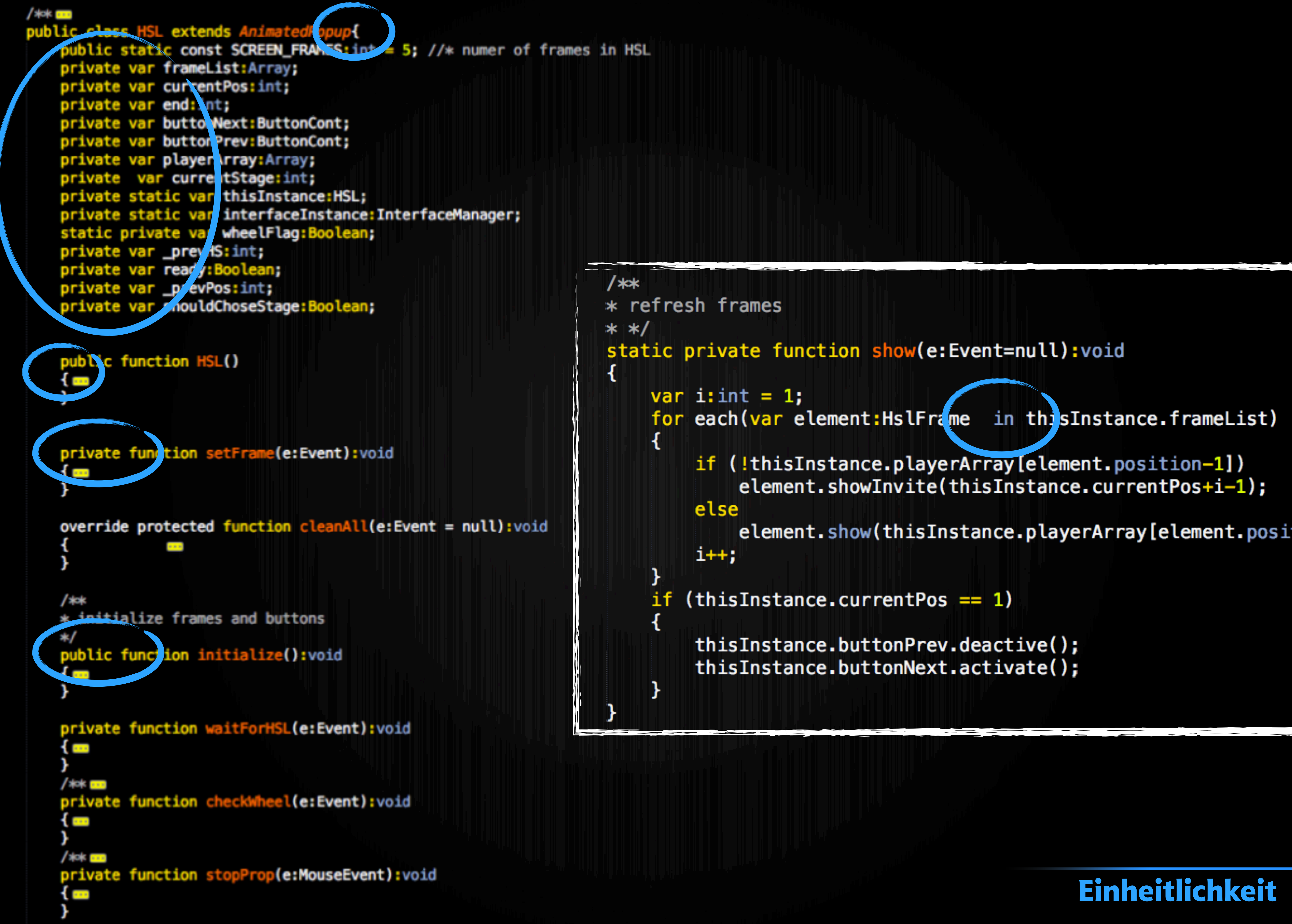
Einheitlichkeit

Einheitlichkeit



```
/** ...
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;

    public function HSL()
    {...
    }

    private function setFrame(e:Event):void
    {...
    }

    override protected function cleanAll(e:Event = null):void
    {            ...
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {...
    }

    private function waitForHSL(e:Event):void
    {...
    }
    /** ...
    private function checkWheel(e:Event):void
    {...
    }
    /** ...
    private function stopProp(e:MouseEvent):void
    {...
    }
```

```
/**
* refresh frames
* */
static private function show(e:Event=null):void
{
    var i:int = 1;
    for each(var element:HslFrame  in thisInstance.frameList)
    {
        if (!thisInstance.playerArray[element.position-1])
            element.showInvite(thisInstance.currentPos+i-1);
        else
            element.show(thisInstance.playerArray[element.posi
        i++;
    }
    if (thisInstance.currentPos == 1)
    {
        thisInstance.buttonPrev.deactive();
        thisInstance.buttonNext.activate();
    }
}
```



```
/** ▭
public class HSL extends AnimatedPopup{
    public static const SCREEN_FRAMES:int = 5; //* numer of frames in HSL
    private var frameList:Array;
    private var currentPos:int;
    private var end:int;
    private var buttonNext:ButtonCont;
    private var buttonPrev:ButtonCont;
    private var playerArray:Array;
    private  var currentStage:int;
    private static var thisInstance:HSL;
    private static var interfaceInstance:InterfaceManager;
    static private var wheelFlag:Boolean;
    private var _prevHS:int;
    private var ready:Boolean;
    private var _prevPos:int;
    private var shouldChoseStage:Boolean;

    public function HSL()
    {▭
    }

    private function setFrame(e:Event):void
    {▭
    }

    override protected function cleanAll(e:Event = null):void
    {           ▭
    }

    /**
    * initialize frames and buttons
    */
    public function initialize():void
    {▭
    }

    private function waitForHSL(e:Event):void
    {▭
    }
    /** ▭
    private function checkWheel(e:Event):void
    {▭
    }
    /** ▭
    private function stopProp(e:MouseEvent):void
    {▭
    }
```

```
/**
* refresh frames
* */
static private function show(e:Event=null):void
{
    var i:int = 1;
    for each(var element:HslFrame  in thisInstance.frameList)
    {
        if (!thisInstance.playerArray[element.position-1])
            element.showInvite(thisInstance.currentPos+i-1);
        else
            element.show(thisInstance.playerArray[element.posi
        i++;
    }
    if (thisInstance.currentPos == 1)
    {
        thisInstance.buttonPrev.deactive();
        thisInstance.buttonNext.activate();
    }
}
```

Einheitlichkeit

Formatierung

Formatierung

- Einrückungen

Formatierung

- Einrückungen
- Klammernumbrüche

Formatierung

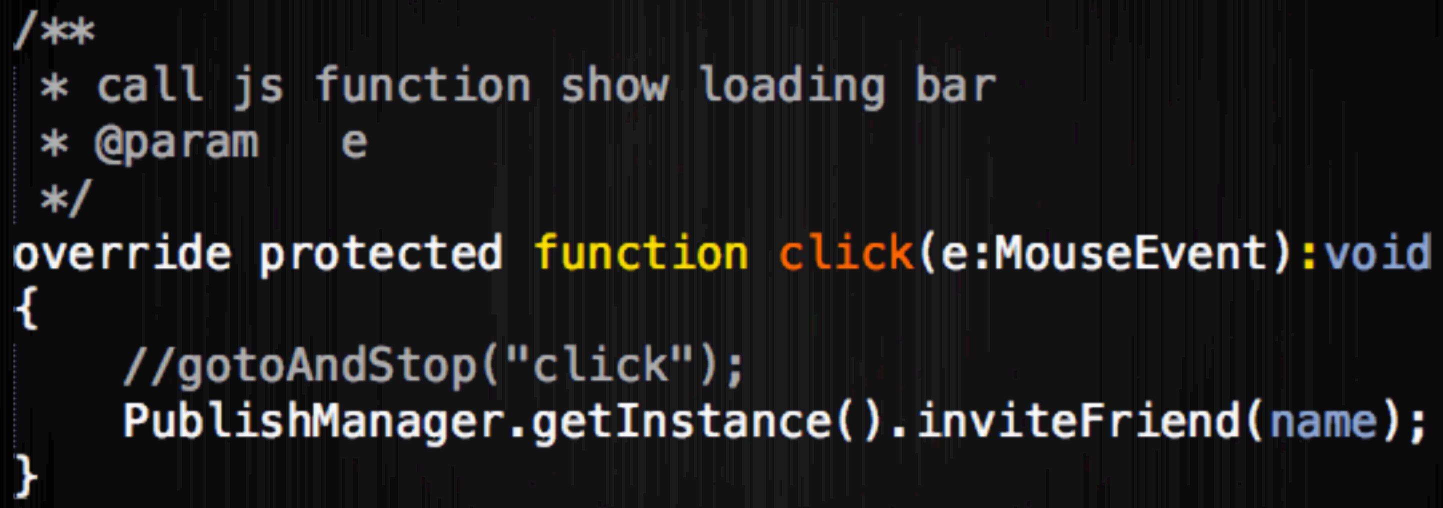
- Einrückungen
- Klammernumbrüche

- Variablen immer zuerst

Formatierung

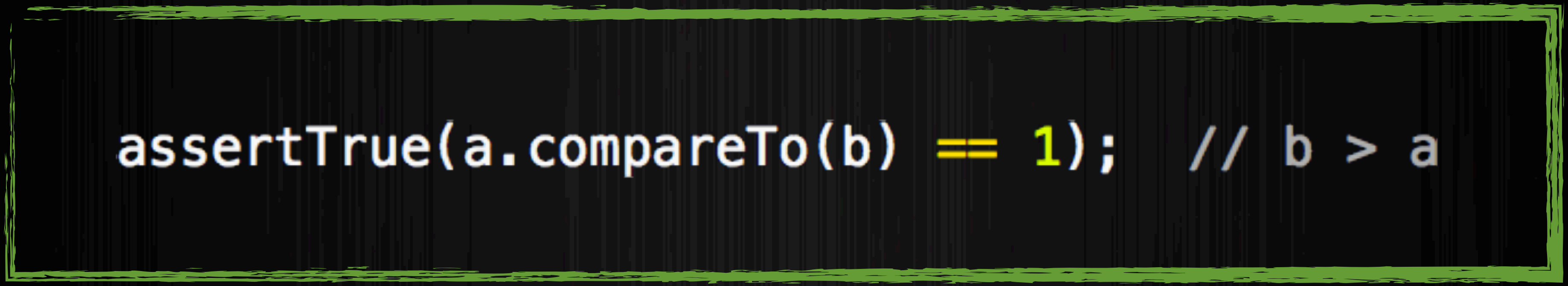
- Einrückungen
- Klammernumbrüche

- Variablen immer zuerst
- Öffentliche Methoden zuerst



```
/**
 * call js function show loading bar
 * @param   e
 */
override protected function click(e:MouseEvent):void
{
    //gotoAndStop("click");
    PublishManager.getInstance().inviteFriend(name);
}
```

Einheitlichkeit



```
assertTrue(a.compareTo(b) == 1);  // b > a
```

Einheitlichkeit

Kommentare

Kommentare

- Vermeiden wenn möglich

Kommentare

- Vermeiden wenn möglich
- Nutzungsszenarien

Kommentare

- Vermeiden wenn möglich
- Nutzungsszenarien
    - Warnungen

Kommentare

- Vermeiden wenn möglich
- Nutzungsszenarien
  - Warnungen
  - Vereinfachte Erklärungen

Einheitlichkeit

Einheitlichkeit

- Namensgebung

Einheitlichkeit

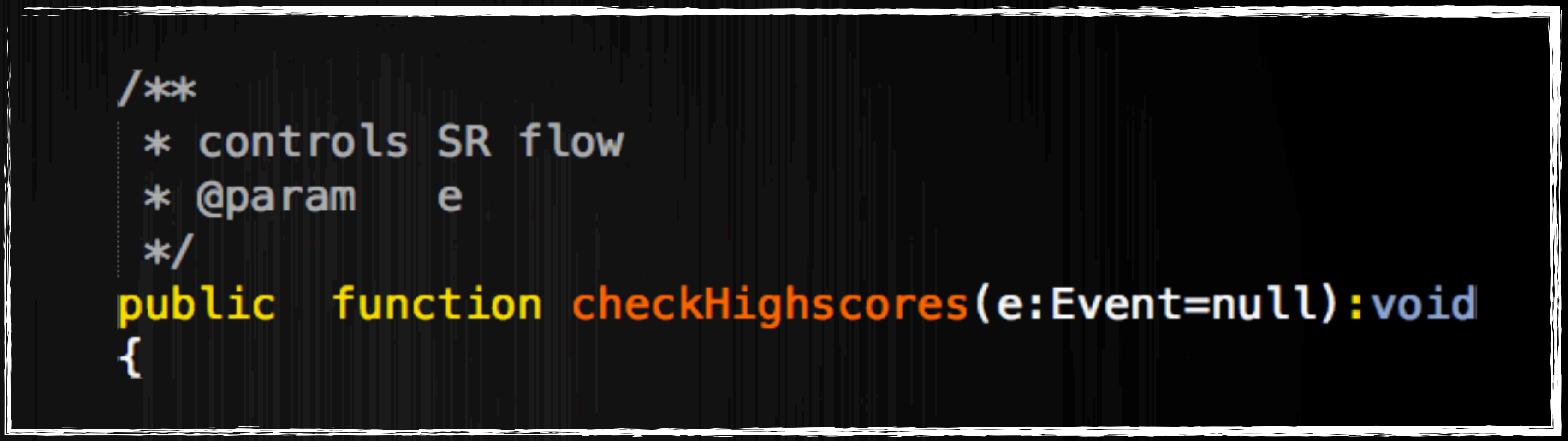
- Namensgebung
- Formatierung

Einheitlichkeit

- Namensgebung
- Formatierung
- **Kommentare**

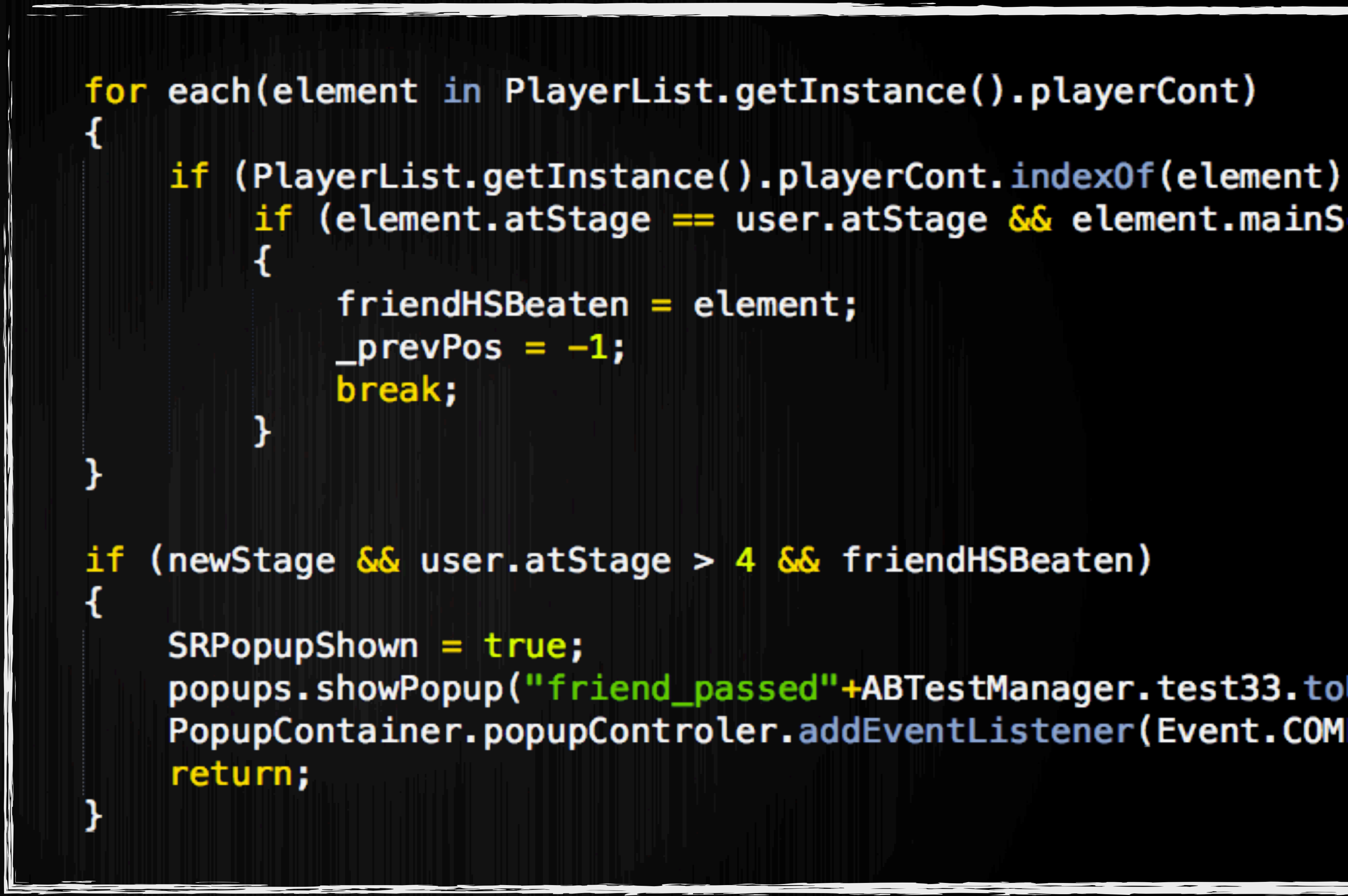
2

Wartbarkeit



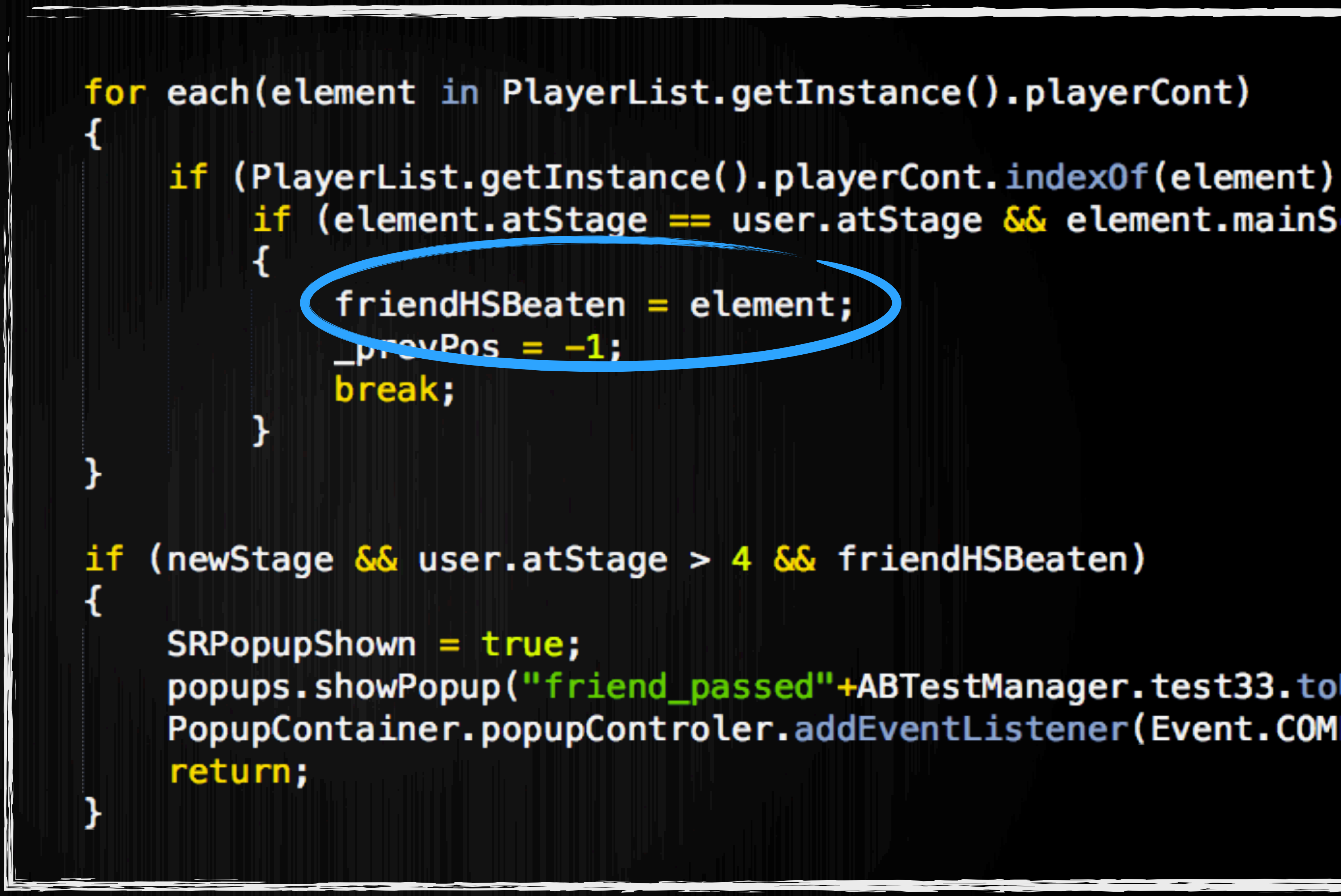
```
/**
 * controls SR flow
 * @param   e
 */
public  function checkHighscores(e:Event=null):void
{
```

Wartbarkeit



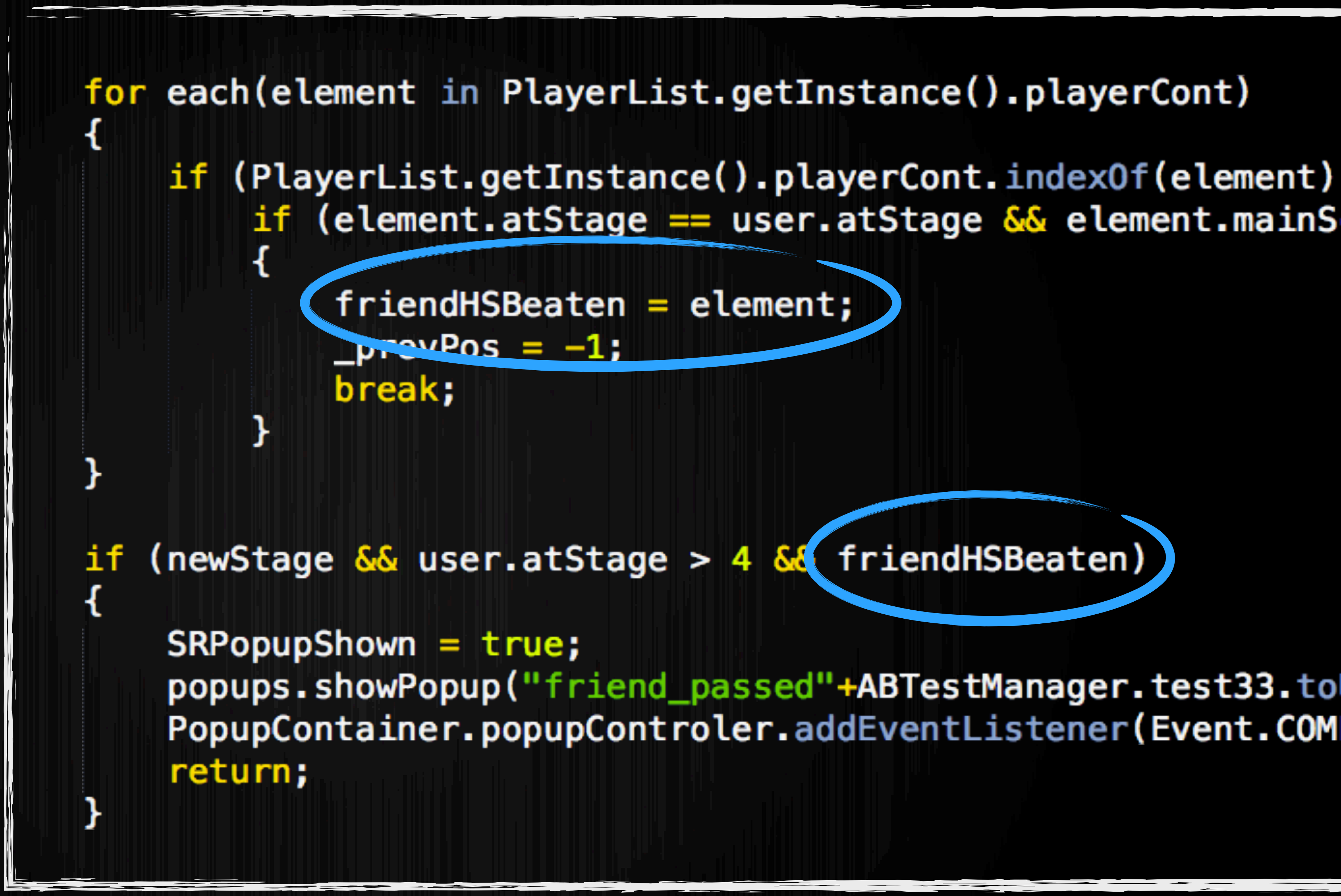
```
for each(element in PlayerList.getInstance().playerCont)
{
    if (PlayerList.getInstance().playerCont.indexOf(element)
        if (element.atStage == user.atStage && element.mainS
        {
            friendHSBeaten = element;
            _prevPos = -1;
            break;
        }
}

if (newStage && user.atStage > 4 && friendHSBeaten)
{
    SRPopupShown = true;
    popups.showPopup("friend_passed"+ABTestManager.test33.to
    PopupContainer.popupControler.addEventListener(Event.COM
    return;
}
```

Wartbarkeit



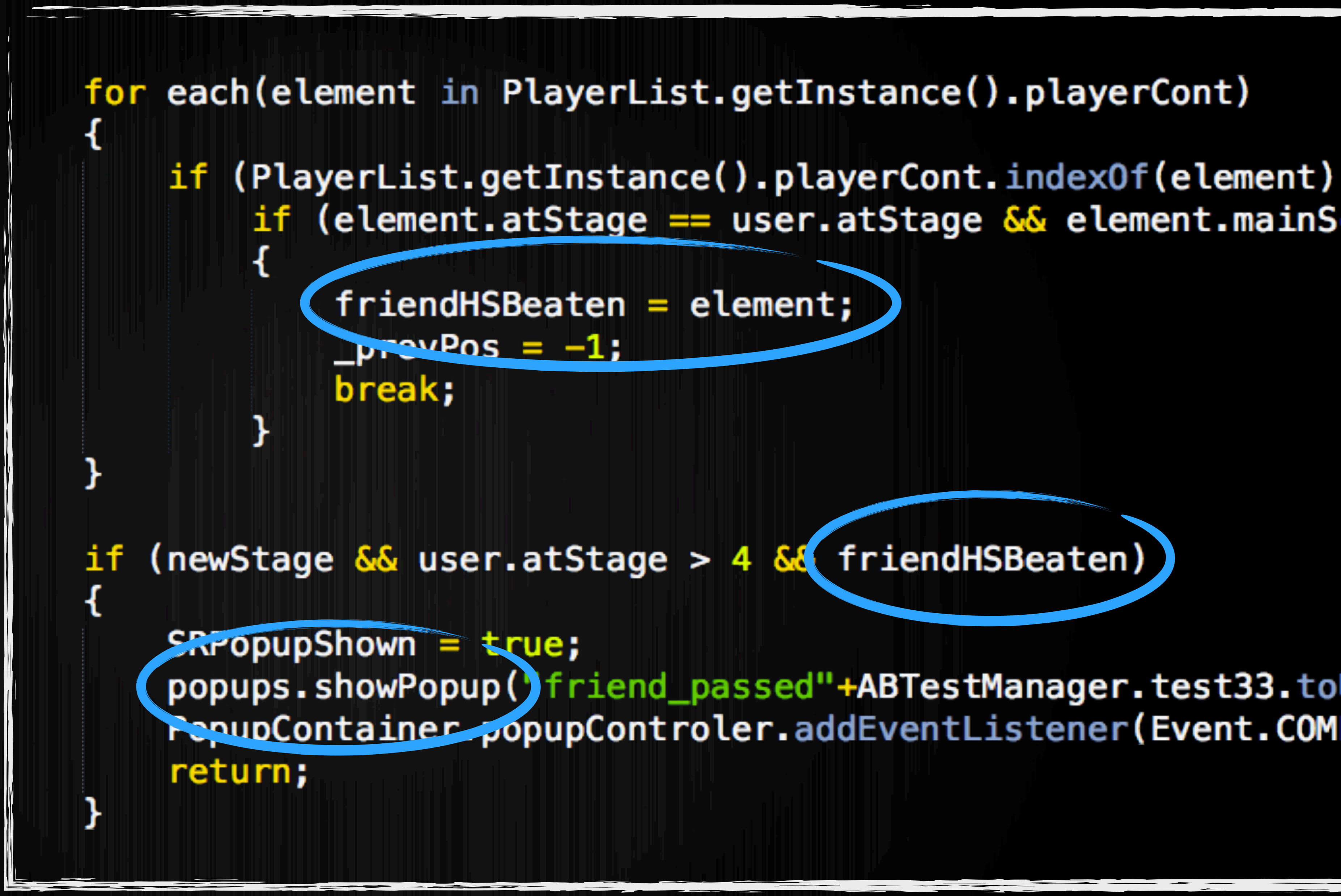
```
for each(element in PlayerList.getInstance().playerCont)
{
    if (PlayerList.getInstance().playerCont.indexOf(element)
        if (element.atStage == user.atStage && element.mainS
        {
            friendHSBeaten = element;
            _prevPos = -1;
            break;
        }
}

if (newStage && user.atStage > 4 && friendHSBeaten)
{
    SRPopupShown = true;
    popups.showPopup("friend_passed"+ABTestManager.test33.to
    PopupContainer.popupControler.addEventListener(Event.COM
    return;
}
```

Wartbarkeit



```
for each(element in PlayerList.getInstance().playerCont)
{
    if (PlayerList.getInstance().playerCont.indexOf(element)
        if (element.atStage == user.atStage && element.mainS
        {
            friendHSBeaten = element;
            _prevPos = -1;
            break;
        }
}

if (newStage && user.atStage > 4 && friendHSBeaten)
{
    SRPopupShown = true;
    popups.showPopup("friend_passed"+ABTestManager.test33.to
    PopupContainer.popupControler.addEventListener(Event.COM
    return;
}
```

Wartbarkeit



```
for each(element in PlayerList.getInstance().playerCont)
{
    if (PlayerList.getInstance().playerCont.indexOf(element)
        if (element.atStage == user.atStage && element.mainS
        {
            friendHSBeaten = element;
            _prevPos = -1;
            break;
        }
}

if (newStage && user.atStage > 4 && friendHSBeaten)
{
    SRPopupShown = true;
    popups.showPopup("friend_passed"+ABTestManager.test33.to
    PopupContainer.popupControler.addEventListener(Event.COM
    return;
}
```

Wartbarkeit

Methoden

Methoden

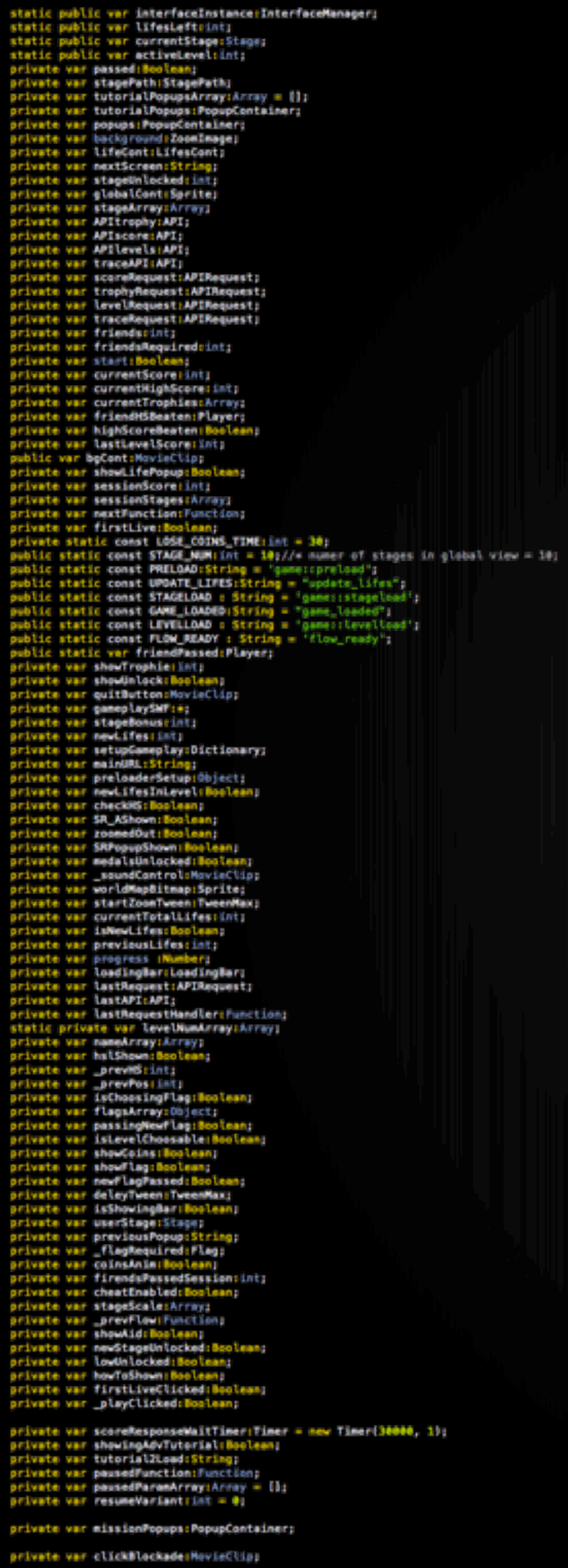
- Möglichst klein

Methoden

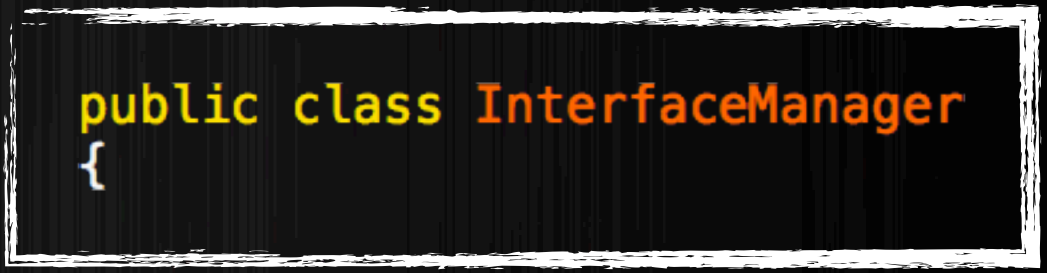
- Möglichst klein
- 1 Abstraktionslevel

Methoden

- Möglichst klein
- 1 Abstraktionslevel
- 1 Funktionalität



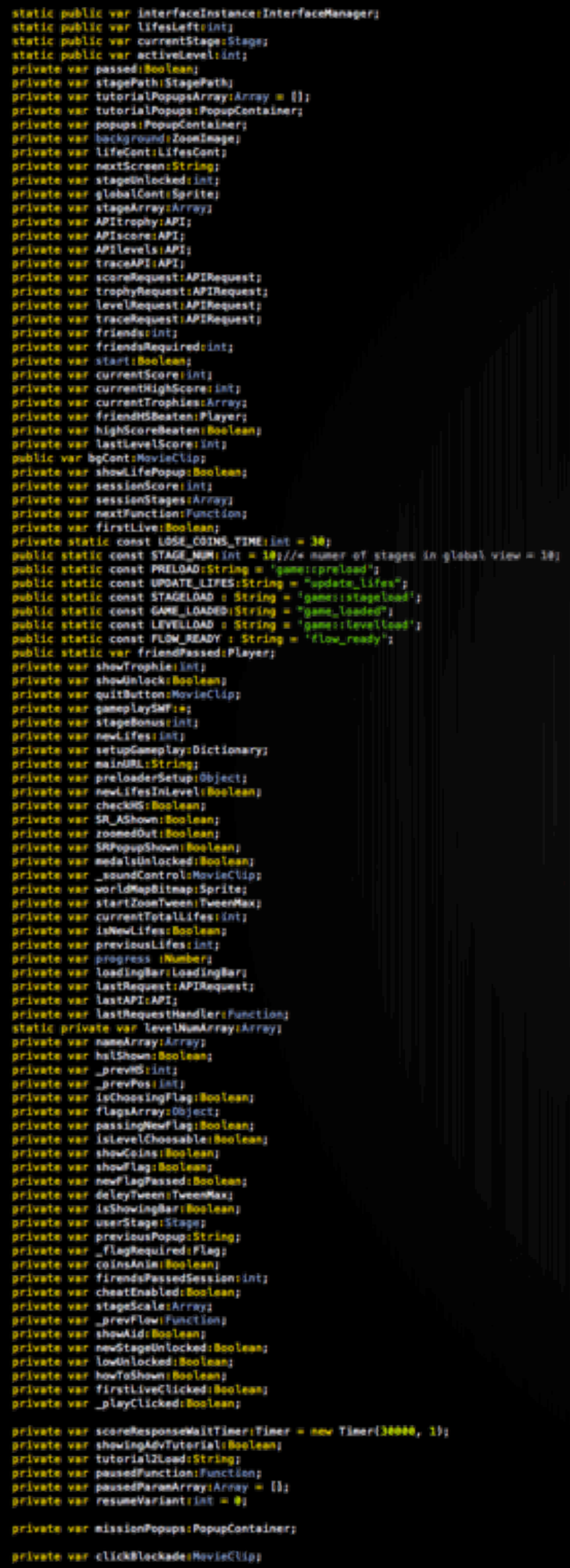
```
static public var interfaceInstance:InterfaceManager;
static public var lifesLeft:int;
static public var currentStage:Stage;
static public var activeLevel:int;
private var passed:Boolean;
private var stagePath:StagePath;
private var tutorialPopupsArray:Array = [];
private var tutorialPopups:PopupContainer;
private var popups:PopupContainer;
private var background:ZoomImage;
private var lifeCont:LifesCont;
private var nextScreen:String;
private var stageUnlocked:int;
private var globalCont:Sprite;
private var stageArray:Array;
private var APItrophy:API;
private var APIscore:API;
private var APIlevels:API;
private var traceAPI:API;
private var scoreRequest:APIRequest;
private var trophyRequest:APIRequest;
private var levelRequest:APIRequest;
private var traceRequest:APIRequest;
private var friends:int;
private var friendsRequired:int;
private var start:Boolean;
private var currentScore:int;
private var currentHighScore:int;
private var currentTrophies:Array;
private var friendHSBeaten:Player;
private var highScoreBeaten:Boolean;
private var lastLevelScore:int;
public var bgCont:MovieClip;
private var showLifePopup:Boolean;
private var sessionScore:int;
private var sessionStages:Array;
private var nextFunction:Function;
private var firstLive:Boolean;
private static const LOSE_COINS_TIME:int = 30;
public static const STAGE_NUM:int = 10;//* numer of stages in global view = 10;
public static const PRELOAD:String = 'game::preload';
public static const UPDATE_LIFES:String = "update_lifes";
public static const STAGELOAD : String = 'game::stageload';
public static const GAME_LOADED:String = "game_loaded";
public static const LEVELLOAD : String = 'game::levelload';
public static const FLOW_READY : String = 'flow_ready';
public static var friendPassed:Player;
private var showTrophie:int;
private var showUnlock:Boolean;
private var quitButton:MovieClip;
private var gameplaySWF:*;
private var stageBonus:int;
private var newLifes:int;
private var setupGameplay:Dictionary;
private var mainURL:String;
private var preloaderSetup:Object;
private var newLifesInLevel:Boolean;
private var checkHS:Boolean;
private var SR_AShown:Boolean;
private var zoomedOut:Boolean;
private var SRPopupShown:Boolean;
private var medalsUnlocked:Boolean;
private var _soundControl:MovieClip;
private var worldMapBitmap:Sprite;
private var startZoomTween:TweenMax;
private var currentTotalLifes:int;
private var isNewLifes:Boolean;
private var previousLifes:int;
private var progress :Number;
private var loadingBar:LoadingBar;
private var lastRequest:APIRequest;
private var lastAPI:API;
private var lastRequestHandler:Function;
static private var levelNumArray:Array;
private var nameArray:Array;
private var hslShown:Boolean;
private var _prevHS:int;
private var _prevPos:int;
private var isChoosingFlag:Boolean;
private var flagsArray:Object;
private var passingNewFlag:Boolean;
private var isLevelChoosable:Boolean;
private var showCoins:Boolean;
private var showFlag:Boolean;
private var newFlagPassed:Boolean;
private var deleyTween:TweenMax;
private var isShowingBar:Boolean;
private var userStage:Stage;
private var previousPopup:String;
private var _flagRequired:Flag;
private var coinsAnim:Boolean;
private var firendsPassedSession:int;
private var cheatEnabled:Boolean;
private var stageScale:Array;
private var _prevFlow:Function;
private var showAid:Boolean;
private var newStageUnlocked:Boolean;
private var lowUnlocked:Boolean;
private var howToShown:Boolean;
private var firstLiveClicked:Boolean;
private var _playClicked:Boolean;

private var scoreResponseWaitTimer:Timer = new Timer(30000, 1);
private var showingAdvTutorial:Boolean;
private var tutorial2Load:String;
private var pausedFunction:Function;
private var pausedParamArray:Array = [];
private var resumeVariant:int = 0;

private var missionPopups:PopupContainer;

private var clickBlockade:MovieClip;
```



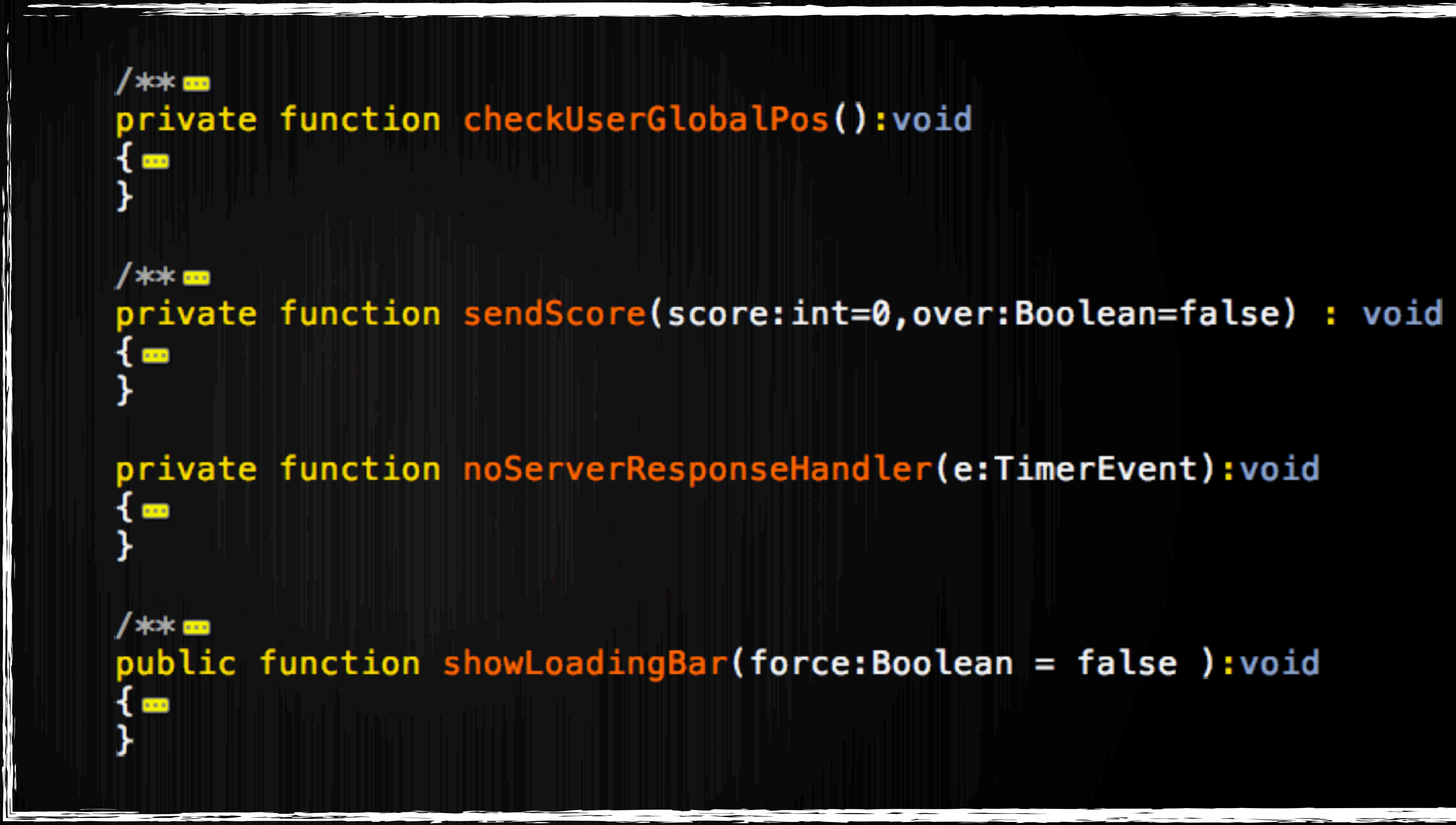
```
public class InterfaceManager
{
```

Wartbarkeit

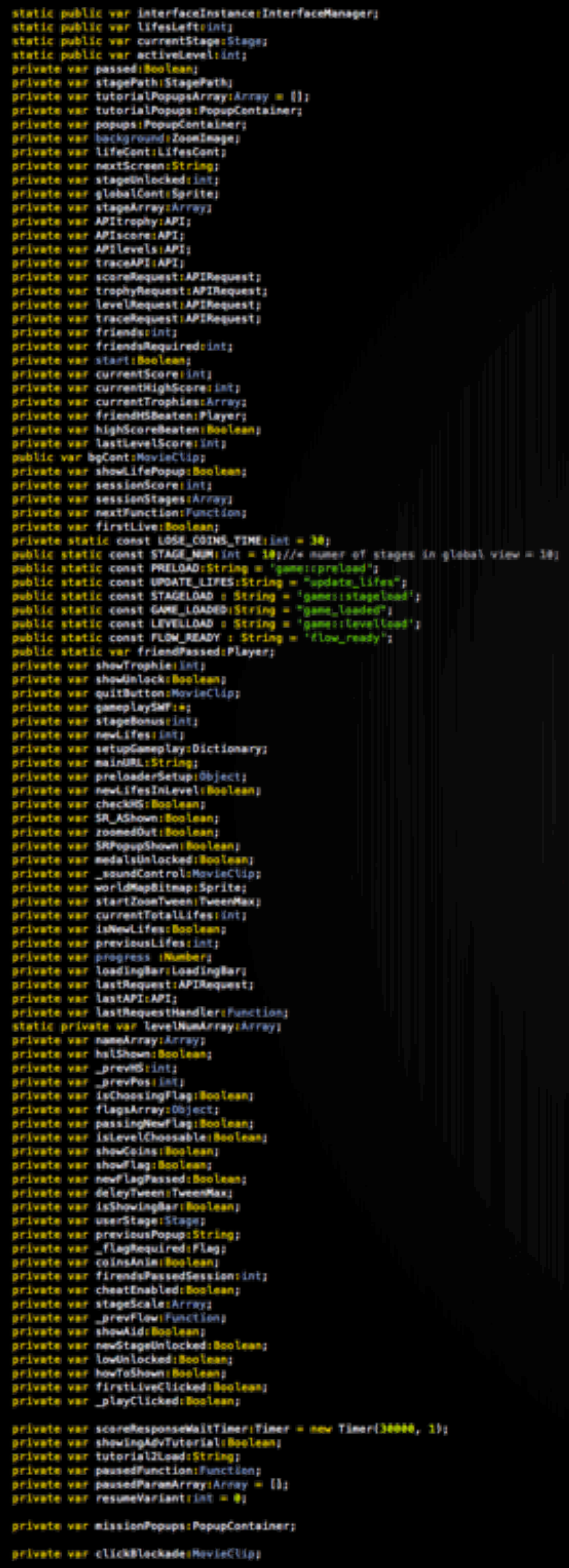
Wartbarkeit



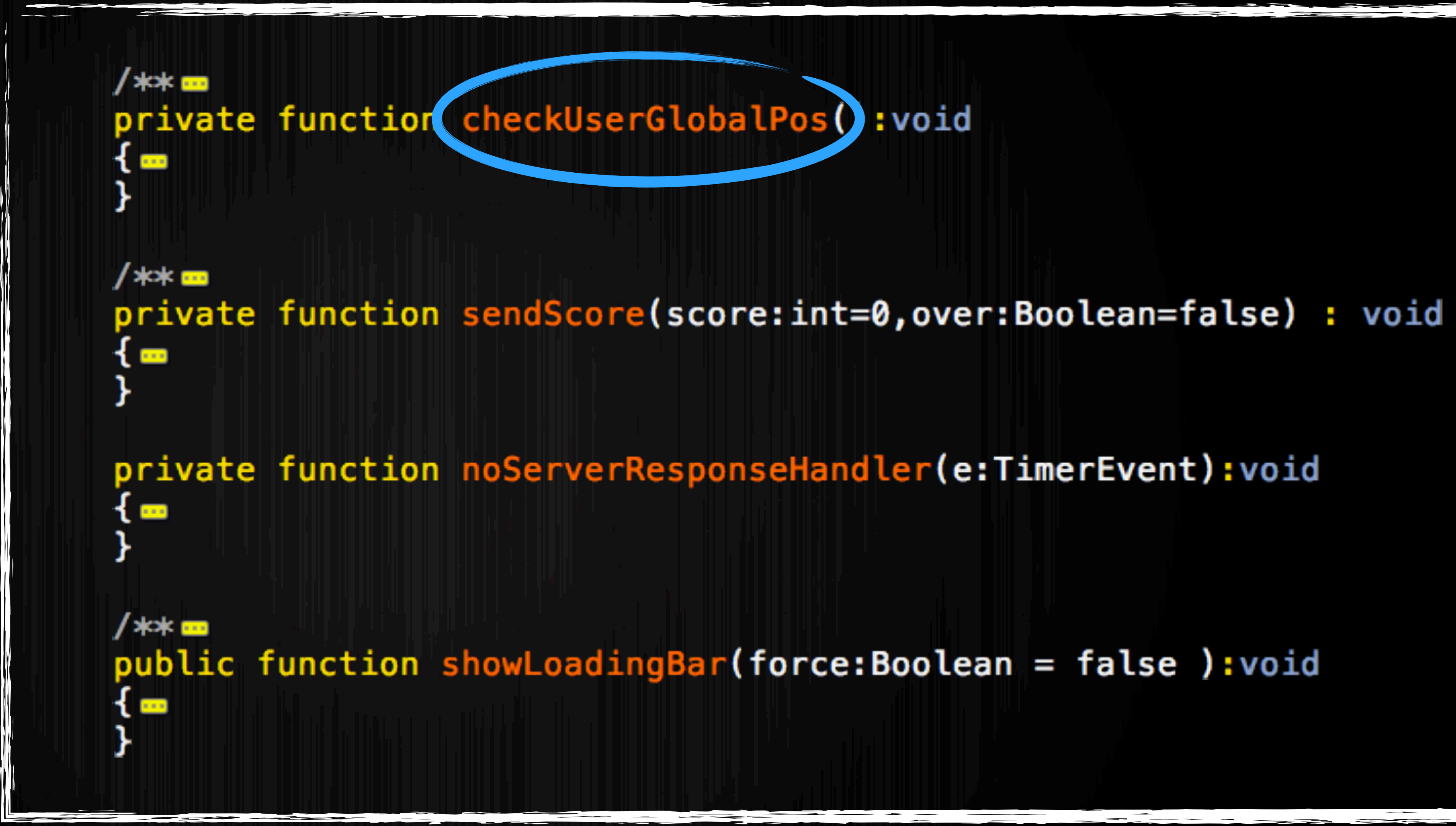
```
static public var interfaceInstance:InterfaceManager;
static public var lifesLeft:int;
static public var currentStage:Stage;
static public var activeLevel:int;
private var passed:Boolean;
private var stagePath:StagePath;
private var tutorialPopupsArray:Array = [];
private var tutorialPopups:PopupContainer;
private var popups:PopupContainer;
private var background:ZoomImage;
private var lifeCont:LifesCont;
private var nextScreen:String;
private var stageUnlocked:int;
private var globalCont:Sprite;
private var stageArray:Array;
private var APItrophy:API;
private var APIscore:API;
private var APIlevels:API;
private var traceAPI:API;
private var scoreRequest:APIRequest;
private var trophyRequest:APIRequest;
private var levelRequest:APIRequest;
private var traceRequest:APIRequest;
private var friends:int;
private var friendsRequired:int;
private var start:Boolean;
private var currentScore:int;
private var currentHighScore:int;
private var currentTrophies:Array;
private var friendHSBeaten:Player;
private var highScoreBeaten:Boolean;
private var lastLevelScore:int;
public var bgCont:MovieClip;
private var showLifePopup:Boolean;
private var sessionScore:int;
private var sessionStages:Array;
private var nextFunction:Function;
private var firstLive:Boolean;
private static const LOSE_COINS_TIME:int = 30;
public static const STAGE_NUM:int = 10;//* numer of stages in global view = 10;
public static const PRELOAD:String = 'game::preload';
public static const UPDATE_LIFES:String = "update_lifes";
public static const STAGELOAD : String = 'game::stageload';
public static const GAME_LOADED:String = "game_loaded";
public static const LEVELLOAD : String = 'game::levelload';
public static const FLOW_READY : String = 'flow_ready';
public static var friendPassed:Player;
private var showTrophie:int;
private var showUnlock:Boolean;
private var quitButton:MovieClip;
private var gameplaySWF:*;
private var stageBonus:int;
private var newLifes:int;
private var setupGameplay:Dictionary;
private var mainURL:String;
private var preloaderSetup:Object;
private var newLifesInLevel:Boolean;
private var checkHS:Boolean;
private var SR_AShown:Boolean;
private var zoomedOut:Boolean;
private var SRPopupShown:Boolean;
private var medalsUnlocked:Boolean;
private var _soundControl:MovieClip;
private var worldMapBitmap:Sprite;
private var startZoomTween:TweenMax;
private var currentTotalLifes:int;
private var isNewLifes:Boolean;
private var previousLifes:int;
private var progress :Number;
private var loadingBar:LoadingBar;
private var lastRequest:APIRequest;
private var lastAPI:API;
private var lastRequestHandler:Function;
static private var levelNumArray:Array;
private var nameArray:Array;
private var hslShown:Boolean;
private var _prevHS:int;
private var _prevPos:int;
private var isChoosingFlag:Boolean;
private var flagsArray:Object;
private var passingNewFlag:Boolean;
private var isLevelChoosable:Boolean;
private var showCoins:Boolean;
private var showFlag:Boolean;
private var newFlagPassed:Boolean;
private var deleyTween:TweenMax;
private var isShowingBar:Boolean;
private var userStage:Stage;
private var previousPopup:String;
private var _flagRequired:Flag;
private var coinsAnim:Boolean;
private var firendsPassedSession:int;
private var cheatEnabled:Boolean;
private var stageScale:Array;
private var _prevFlow:Function;
private var showAid:Boolean;
private var newStageUnlocked:Boolean;
private var lowUnlocked:Boolean;
private var howToShown:Boolean;
private var firstLiveClicked:Boolean;
private var _playClicked:Boolean;

private var scoreResponseWaitTimer:Timer = new Timer(30000, 1);
private var showingAdvTutorial:Boolean;
private var tutorial2Load:String;
private var pausedFunction:Function;
private var pausedParamArray:Array = [];
private var resumeVariant:int = 0;

private var missionPopups:PopupContainer;

private var clickBlockade:MovieClip;
```



```
/** ...
private function checkUserGlobalPos():void
{ ...
}

/** ...
private function sendScore(score:int=0,over:Boolean=false) : void
{ ...
}

private function noServerResponseHandler(e:TimerEvent):void
{ ...
}

/** ...
public function showLoadingBar(force:Boolean = false ):void
{ ...
}
```



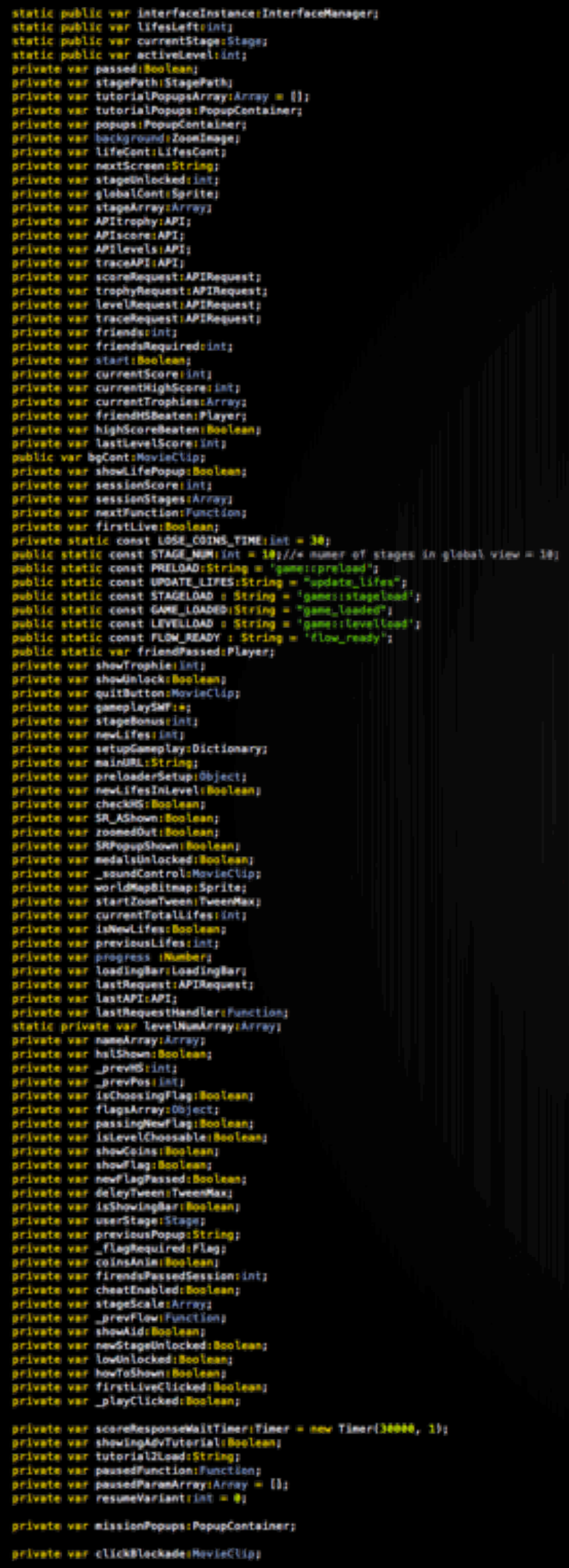
```
static public var interfaceInstance:InterfaceManager;
static public var lifesLeft:int;
static public var currentStage:Stage;
static public var activeLevel:int;
private var passed:Boolean;
private var stagePath:StagePath;
private var tutorialPopupsArray:Array = [];
private var tutorialPopups:PopupContainer;
private var popups:PopupContainer;
private var background:ZoomImage;
private var lifeCont:LifesCont;
private var nextScreen:String;
private var stageUnlocked:int;
private var globalCont:Sprite;
private var stageArray:Array;
private var APItrophy:API;
private var APIscore:API;
private var APIlevels:API;
private var traceAPI:API;
private var scoreRequest:APIRequest;
private var trophyRequest:APIRequest;
private var levelRequest:APIRequest;
private var traceRequest:APIRequest;
private var friends:int;
private var friendsRequired:int;
private var start:Boolean;
private var currentScore:int;
private var currentHighScore:int;
private var currentTrophies:Array;
private var friendHSBeaten:Player;
private var highScoreBeaten:Boolean;
private var lastLevelScore:int;
public var bgCont:MovieClip;
private var showLifePopup:Boolean;
private var sessionScore:int;
private var sessionStages:Array;
private var nextFunction:Function;
private var firstLive:Boolean;
private static const LOSE_COINS_TIME:int = 30;
public static const STAGE_NUM:int = 10;//* numer of stages in global view = 10;
public static const PRELOAD:String = 'game::preload';
public static const UPDATE_LIFES:String = "update_lifes";
public static const STAGELOAD : String = 'game::stageload';
public static const GAME_LOADED:String = "game_loaded";
public static const LEVELLOAD : String = 'game::levelload';
public static const FLOW_READY : String = 'flow_ready';
public static var friendPassed:Player;
private var showTrophie:int;
private var showUnlock:Boolean;
private var quitButton:MovieClip;
private var gameplaySWF:*;
private var stageBonus:int;
private var newLifes:int;
private var setupGameplay:Dictionary;
private var mainURL:String;
private var preloaderSetup:Object;
private var newLifesInLevel:Boolean;
private var checkHS:Boolean;
private var SR_AShown:Boolean;
private var zoomedOut:Boolean;
private var SRPopupShown:Boolean;
private var medalsUnlocked:Boolean;
private var _soundControl:MovieClip;
private var worldMapBitmap:Sprite;
private var startZoomTween:TweenMax;
private var currentTotalLifes:int;
private var isNewLifes:Boolean;
private var previousLifes:int;
private var progress :Number;
private var loadingBar:LoadingBar;
private var lastRequest:APIRequest;
private var lastAPI:API;
private var lastRequestHandler:Function;
static private var levelNumArray:Array;
private var nameArray:Array;
private var hslShown:Boolean;
private var _prevHS:int;
private var _prevPos:int;
private var isChoosingFlag:Boolean;
private var flagsArray:Object;
private var passingNewFlag:Boolean;
private var isLevelChoosable:Boolean;
private var showCoins:Boolean;
private var showFlag:Boolean;
private var newFlagPassed:Boolean;
private var deleyTween:TweenMax;
private var isShowingBar:Boolean;
private var userStage:Stage;
private var previousPopup:String;
private var _flagRequired:Flag;
private var coinsAnim:Boolean;
private var firendsPassedSession:int;
private var cheatEnabled:Boolean;
private var stageScale:Array;
private var _prevFlow:Function;
private var showAid:Boolean;
private var newStageUnlocked:Boolean;
private var lowUnlocked:Boolean;
private var howToShown:Boolean;
private var firstLiveClicked:Boolean;
private var _playClicked:Boolean;

private var scoreResponseWaitTimer:Timer = new Timer(30000, 1);
private var showingAdvTutorial:Boolean;
private var tutorial2Load:String;
private var pausedFunction:Function;
private var pausedParamArray:Array = [];
private var resumeVariant:int = 0;

private var missionPopups:PopupContainer;

private var clickBlockade:MovieClip;
```



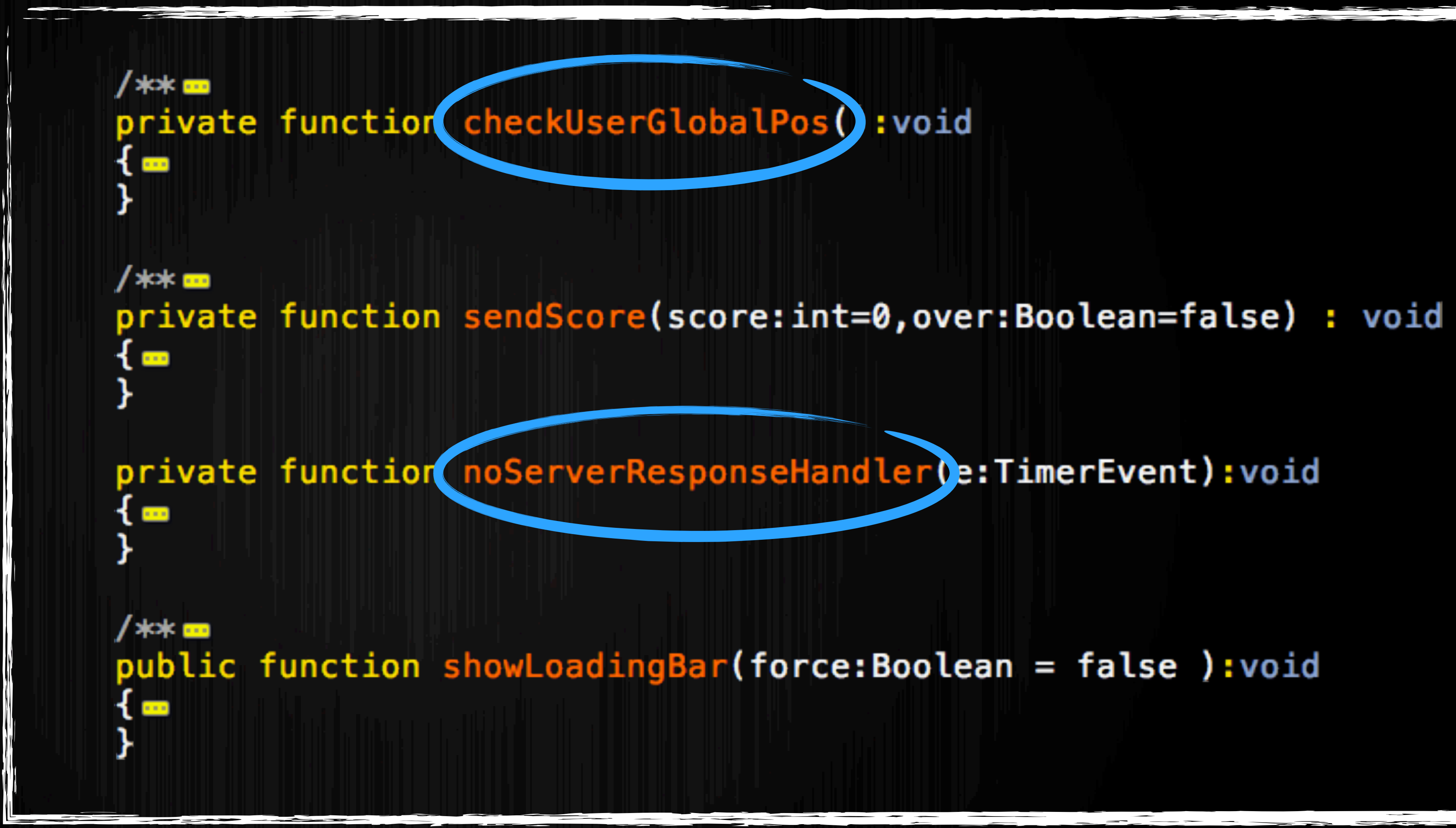
```
/** ...
private function checkUserGlobalPos( ):void
{ ...
}

/** ...
private function sendScore(score:int=0,over:Boolean=false) : void
{ ...
}

private function noServerResponseHandler(e:TimerEvent):void
{ ...
}

/** ...
public function showLoadingBar(force:Boolean = false ):void
{ ...
}
```

Wartbarkeit

Wartbarkeit



```
static public var interfaceInstance:InterfaceManager;
static public var lifesLeft:int;
static public var currentStage:Stage;
static public var activeLevel:int;
private var passed:Boolean;
private var stagePath:StagePath;
private var tutorialPopupsArray:Array = [];
private var tutorialPopups:PopupContainer;
private var popups:PopupContainer;
private var background:ZoomImage;
private var lifeCont:LifesCont;
private var nextScreen:String;
private var stageUnlocked:int;
private var globalCont:Sprite;
private var stageArray:Array;
private var APItrophy:API;
private var APIscore:API;
private var APIlevels:API;
private var traceAPI:API;
private var scoreRequest:APIRequest;
private var trophyRequest:APIRequest;
private var levelRequest:APIRequest;
private var traceRequest:APIRequest;
private var friends:int;
private var friendsRequired:int;
private var start:Boolean;
private var currentScore:int;
private var currentHighScore:int;
private var currentTrophies:Array;
private var friendHSBeaten:Player;
private var highScoreBeaten:Boolean;
private var lastLevelScore:int;
public var bgCont:MovieClip;
private var showLifePopup:Boolean;
private var sessionScore:int;
private var sessionStages:Array;
private var nextFunction:Function;
private var firstLive:Boolean;
private static const LOSE_COINS_TIME:int = 30;
public static const STAGE_NUM:int = 10;//* numer of stages in global view = 10;
public static const PRELOAD:String = 'game::preload';
public static const UPDATE_LIFES:String = "update_lifes";
public static const STAGELOAD : String = 'game::stageload';
public static const GAME_LOADED:String = "game_loaded";
public static const LEVELLOAD : String = 'game::levelload';
public static const FLOW_READY : String = 'flow_ready';
public static var friendPassed:Player;
private var showTrophie:int;
private var showUnlock:Boolean;
private var quitButton:MovieClip;
private var gameplaySWF:*;
private var stageBonus:int;
private var newLifes:int;
private var setupGameplay:Dictionary;
private var mainURL:String;
private var preloaderSetup:Object;
private var newLifesInLevel:Boolean;
private var checkHS:Boolean;
private var SR_AShown:Boolean;
private var zoomedOut:Boolean;
private var SRPopupShown:Boolean;
private var medalsUnlocked:Boolean;
private var _soundControl:MovieClip;
private var worldMapBitmap:Sprite;
private var startZoomTween:TweenMax;
private var currentTotalLifes:int;
private var isNewLifes:Boolean;
private var previousLifes:int;
private var progress :Number;
private var loadingBar:LoadingBar;
private var lastRequest:APIRequest;
private var lastAPI:API;
private var lastRequestHandler:Function;
static private var levelNumArray:Array;
private var nameArray:Array;
private var hslShown:Boolean;
private var _prevHS:int;
private var _prevPos:int;
private var isChoosingFlag:Boolean;
private var flagsArray:Object;
private var passingNewFlag:Boolean;
private var isLevelChoosable:Boolean;
private var showCoins:Boolean;
private var showFlag:Boolean;
private var newFlagPassed:Boolean;
private var deleyTween:TweenMax;
private var isShowingBar:Boolean;
private var userStage:Stage;
private var previousPopup:String;
private var _flagRequired:Flag;
private var coinsAnim:Boolean;
private var firendsPassedSession:int;
private var cheatEnabled:Boolean;
private var stageScale:Array;
private var _prevFlow:Function;
private var showAid:Boolean;
private var newStageUnlocked:Boolean;
private var lowUnlocked:Boolean;
private var howToShown:Boolean;
private var firstLiveClicked:Boolean;
private var _playClicked:Boolean;

private var scoreResponseWaitTimer:Timer = new Timer(30000, 1);
private var showingAdvTutorial:Boolean;
private var tutorial2Load:String;
private var pausedFunction:Function;
private var pausedParamArray:Array = [];
private var resumeVariant:int = 0;

private var missionPopups:PopupContainer;

private var clickBlockade:MovieClip;
```



```
/** ...
private function checkUserGlobalPos( ):void
{ ...
}

/** ...
private function sendScore(score:int=0,over:Boolean=false) : void
{ ...
}

private function noServerResponseHandler(e:TimerEvent):void
{ ...
}

/** ...
public function showLoadingBar(force:Boolean = false ):void
{ ...
}
```

Klassen

Klassen

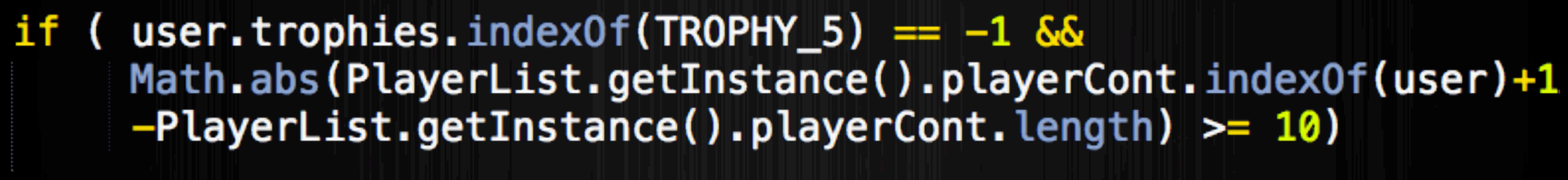
- 1 Grund für Veränderung (SRP)

Klassen

- 1 Grund für Veränderung (SRP)
- Wenige Instanzvariablen

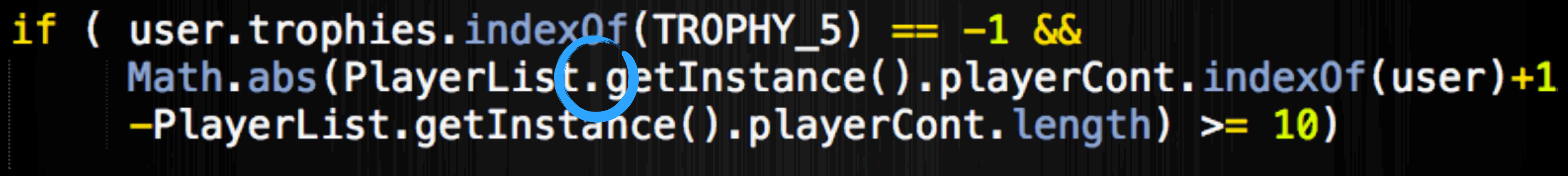
Klassen

- 1 Grund für Veränderung (SRP)
- Wenige Instanzvariablen
- Hohe Kohesion



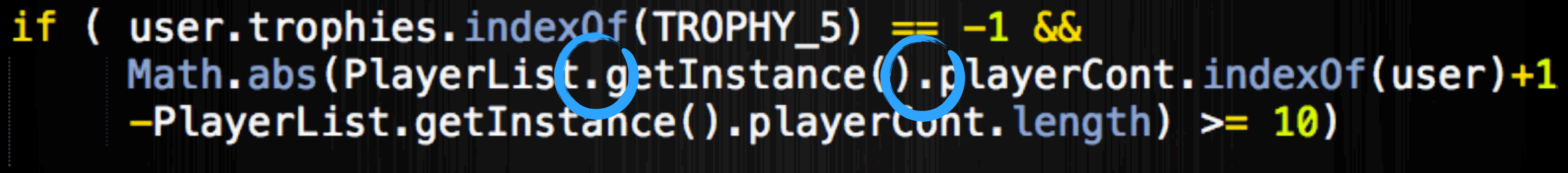
```
if ( user.trophies.indexOf(TROPHY_5) == -1 &&
     Math.abs(PlayerList.getInstance().playerCont.indexOf(user)+1
     -PlayerList.getInstance().playerCont.length) >= 10)
```

Wartbarkeit



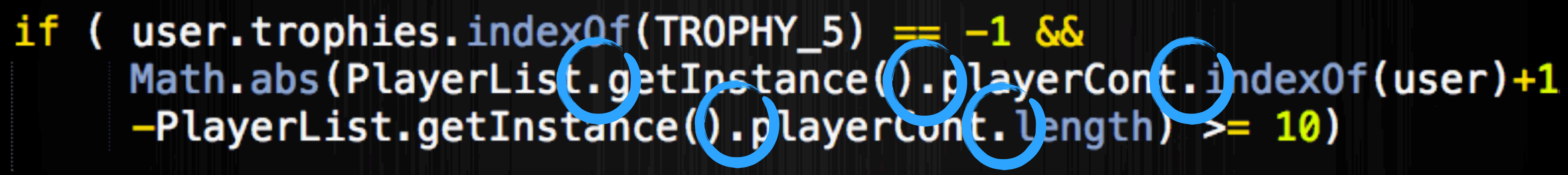
```
if ( user.trophies.indexOf(TROPHY_5) == -1 &&
     Math.abs(PlayerList.getInstance().playerCont.indexOf(user)+1
     -PlayerList.getInstance().playerCont.length) >= 10)
```

Wartbarkeit



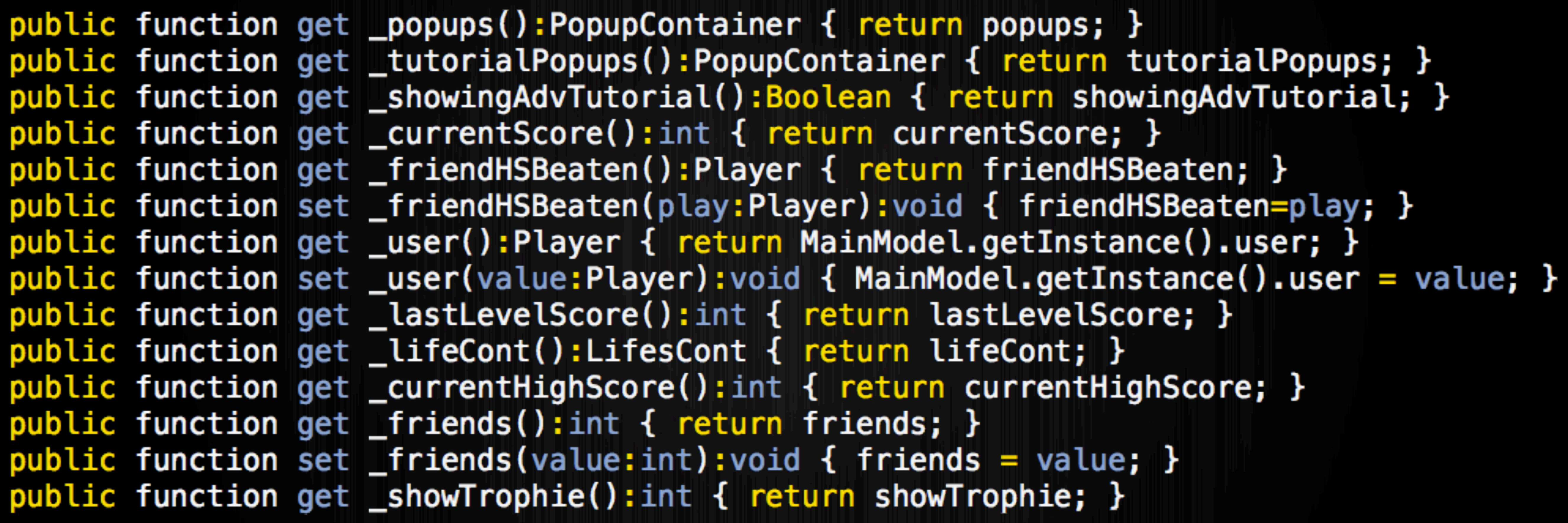
```
if ( user.trophies.indexOf(TROPHY_5) == -1 &&
     Math.abs(PlayerList.getInstance().playerCont.indexOf(user)+1
     -PlayerList.getInstance().playerCont.length) >= 10)
```

Wartbarkeit



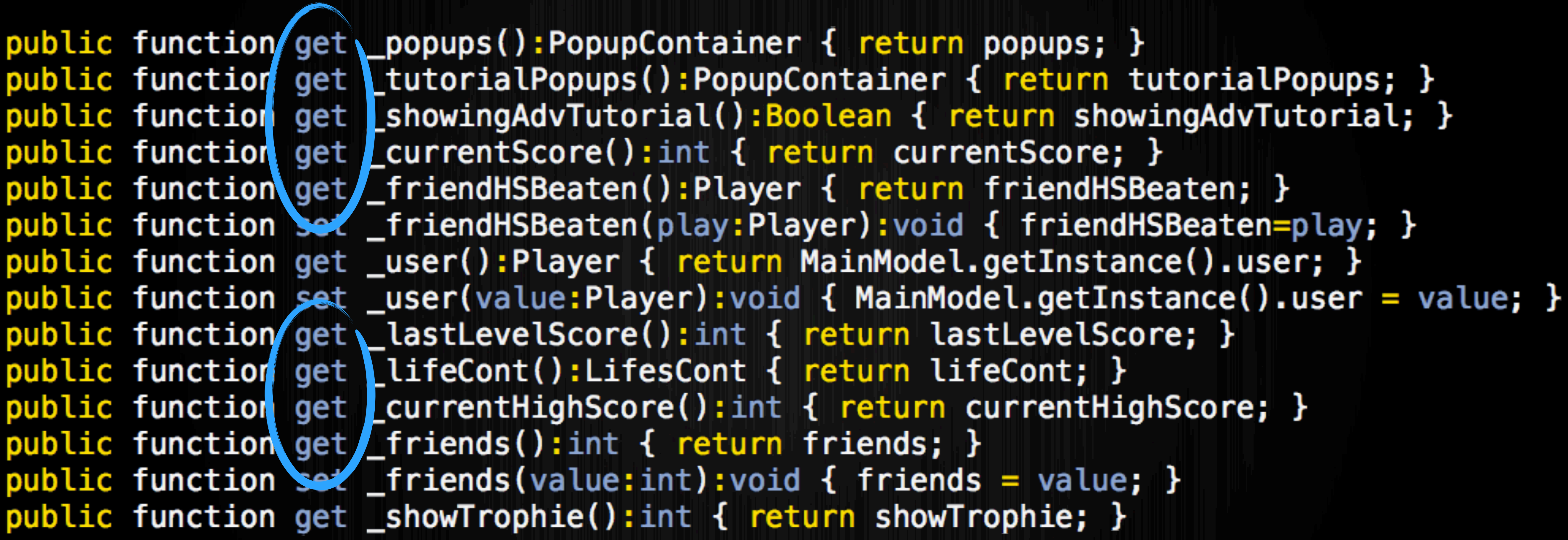
```
if ( user.trophies.indexOf(TROPHY_5) == -1 &&
     Math.abs(PlayerList.getInstance().playerCont.indexOf(user)+1
     -PlayerList.getInstance().playerCont.length) >= 10)
```

Wartbarkeit



```
public function get _popups():PopupContainer { return popups; }
public function get _tutorialPopups():PopupContainer { return tutorialPopups; }
public function get _showingAdvTutorial():Boolean { return showingAdvTutorial; }
public function get _currentScore():int { return currentScore; }
public function get _friendHSBeaten():Player { return friendHSBeaten; }
public function set _friendHSBeaten(play:Player):void { friendHSBeaten=play; }
public function get _user():Player { return MainModel.getInstance().user; }
public function set _user(value:Player):void { MainModel.getInstance().user = value; }
public function get _lastLevelScore():int { return lastLevelScore; }
public function get _lifeCont():LifesCont { return lifeCont; }
public function get _currentHighScore():int { return currentHighScore; }
public function get _friends():int { return friends; }
public function set _friends(value:int):void { friends = value; }
public function get _showTrophie():int { return showTrophie; }
```

Wartbarkeit



```
public function get _popups():PopupContainer { return popups; }
public function get _tutorialPopups():PopupContainer { return tutorialPopups; }
public function get _showingAdvTutorial():Boolean { return showingAdvTutorial; }
public function get _currentScore():int { return currentScore; }
public function get _friendHSBeaten():Player { return friendHSBeaten; }
public function set _friendHSBeaten(play:Player):void { friendHSBeaten=play; }
public function get _user():Player { return MainModel.getInstance().user; }
public function set _user(value:Player):void { MainModel.getInstance().user = value; }
public function get _lastLevelScore():int { return lastLevelScore; }
public function get _lifeCont():LifesCont { return lifeCont; }
public function get _currentHighScore():int { return currentHighScore; }
public function get _friends():int { return friends; }
public function set _friends(value:int):void { friends = value; }
public function get _showTrophie():int { return showTrophie; }
```

Wartbarkeit

Datenstrukturen

Datenstrukturen

- Train Wrecks vermeiden

Datenstrukturen

- Train Wrecks vermeiden
- Data Transfer Objects nutzen

Datenstrukturen

- Train Wrecks vermeiden
- Data Transfer Objects nutzen
- Erzeugung und Nutzung von Systemen trennen

Wartbarkeit

Wartbarkeit

- Methoden

Wartbarkeit

- Methoden
- Klassen

Wartbarkeit

- Methoden
- Klassen
- Datenstrukturen

3

Agilität

Wissensinseln

http://flickr.com/photos/92928628@N04

Pair Programming

Pair Programming

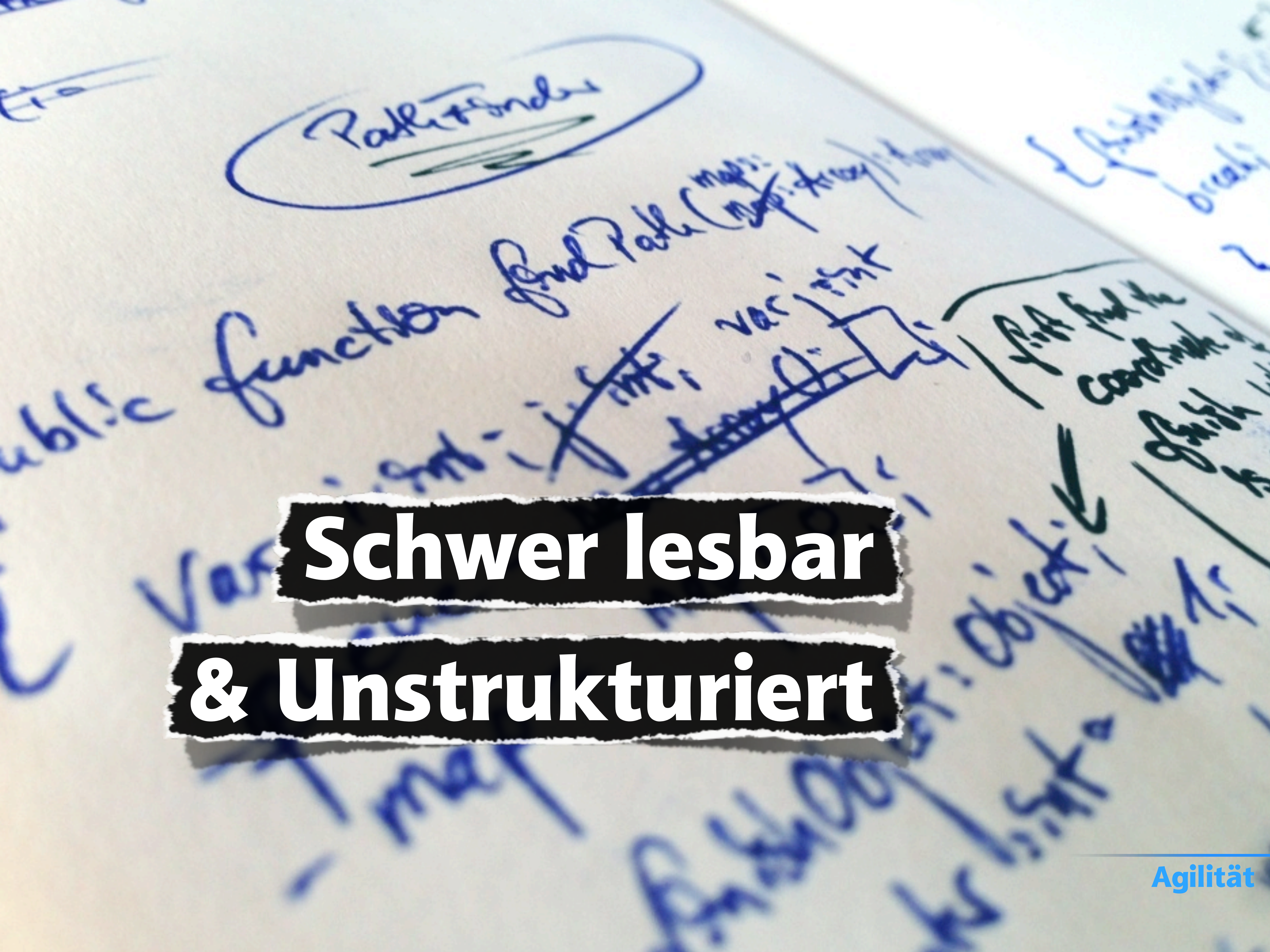
- Vermeiden von Wissensinseln

Pair Programming

- Vermeiden von Wissensinseln
- Integration neuer Teammitglieder

Pair Programming

- Vermeiden von Wissensinseln
- Integration neuer Teammitglieder
- **Kollektive Verantwortung**


PathFinder
public function findPath(maps: Array): Array
var
int
Object

Schwer lesbar
& Unstrukturiert

Regelmäßiges Refactoring

Regelmäßiges Refactoring

- Stetige Verbesserungen

Regelmäßiges Refactoring

- Stetige Verbesserungen
- The Boy Scout Rule (Robert C. Martin)

Keine Absichersichung

http://flickr.com/photos/99995147@N00

Unit-Tests

Unit-Tests

- Sicherheitsnetz

Unit-Tests

- Sicherheitsnetz
- Erleichterte Einarbeitung

Unit-Tests

- Sicherheitsnetz
- Erleichterte Einarbeitung
- **Versuchsfeld**

Features spät testbar

http://www.flickr.com/photos/adesigna

Kontinuierliche Integration

Kontinuierliche Integration

- Softwareartefakte früh nutzbar

Kontinuierliche Integration

- Softwareartefakte früh nutzbar
- Erwartungen überprüfbar

Kontinuierliche Integration

- Softwareartefakte früh nutzbar
- Erwartungen überprüfbar
- Probleme schnell absehbar

Agilität

Agilität

- Pair Programming

Agilität

- Pair Programming
- Regelmäßiges Refactoring

Agilität

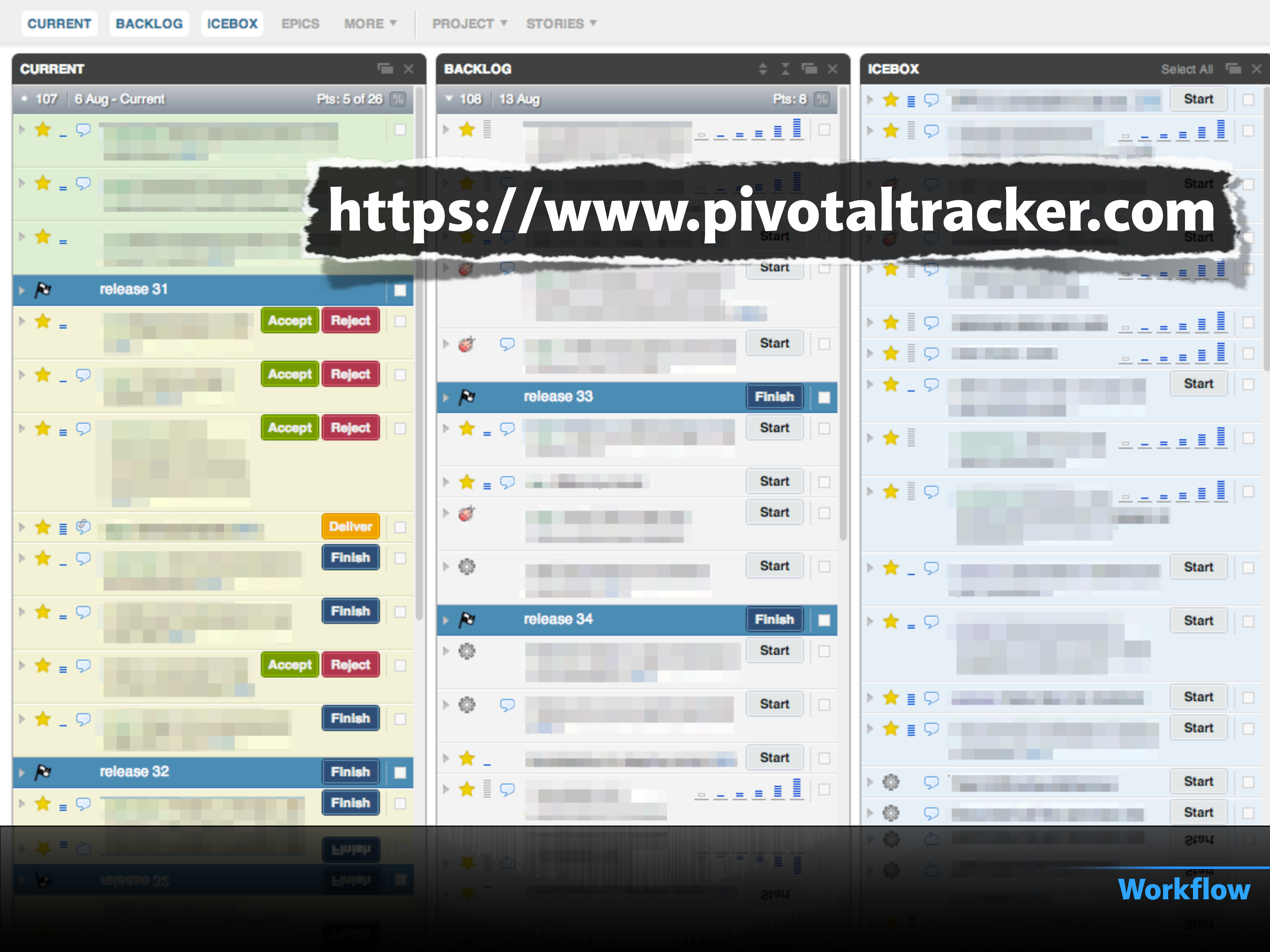
- Pair Programming
- Regelmäßiges Refactoring
- **Unit-Tests**

Agilität

- Pair Programming
- Regelmäßiges Refactoring
- Unit-Tests
- **Kontinuierliche Integration**

4

Werkzeuge

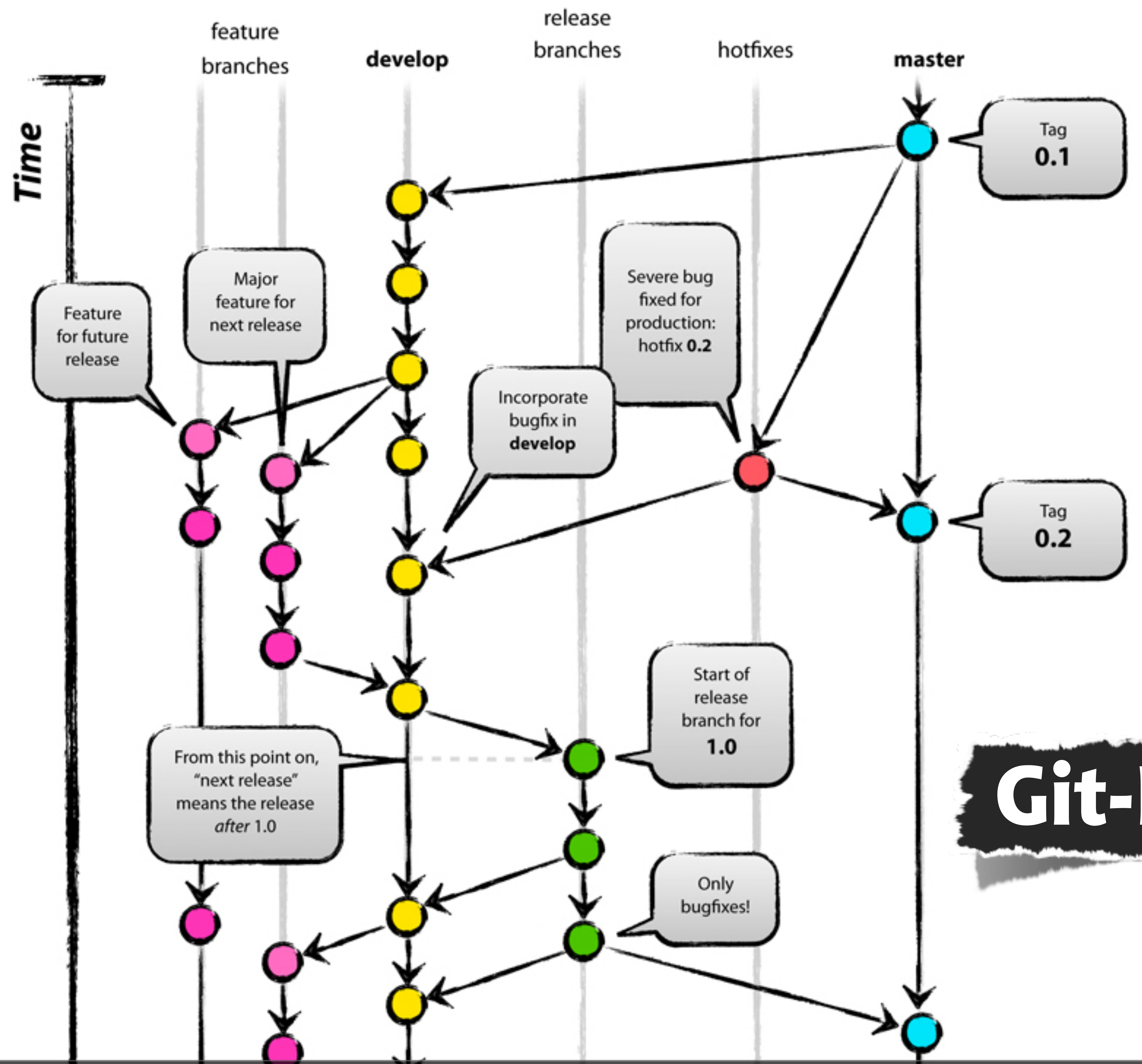


CURRENT
BACKLOG
ICEBOX
EPICS
MORE ▾
PROJECT ▾
STORIES ▾
CURRENT
107
6 Aug - Current
Pts: 5 of 26
release 31
Accept
Reject
Deliver
Finish
release 32
BACKLOG
108
13 Aug
Pts: 8
Start
release 33
release 34
ICEBOX
Select All

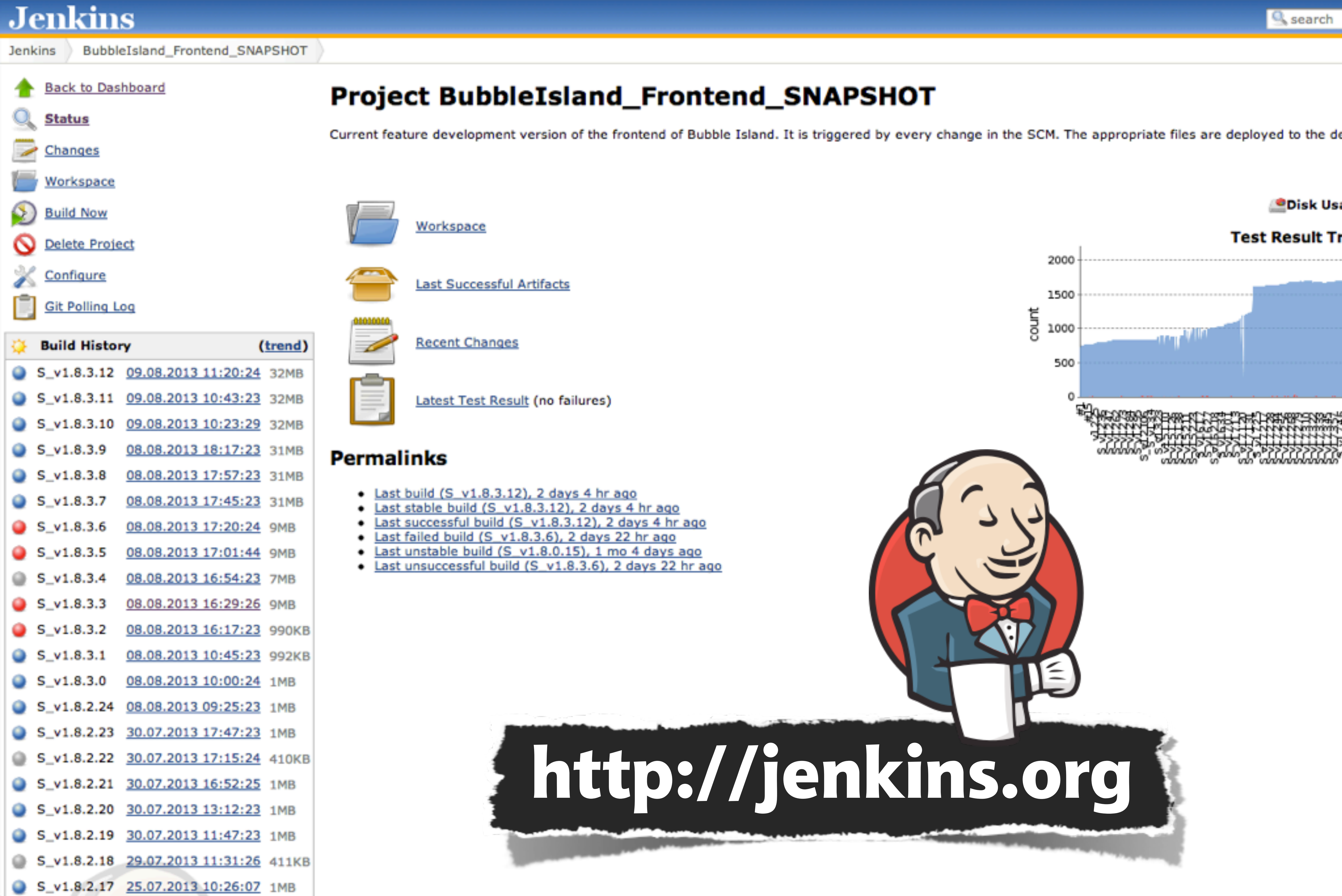
https://www.pivotaltracker.com

Workflow

Git-Flow


Time
feature branches
develop
release branches
hotfixes
master
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix **0.2**
Incorporate bugfix in **develop**
Tag **0.2**
Start of release branch for **1.0**
From this point on, "next release" means the release *after* 1.0
Only bugfixes!

http://nvie.com/img/2009/12/Screen-shot-2009-12-24-at-11.32.03.png


Jenkins
search
Jenkins
BubbleIsland_Frontend_SNAPSHOT
Back to Dashboard
Status
Changes
Workspace
Build Now
Delete Project
Configure
Git Polling Log
Disk Usa
Test Result Tr
count
2000
1500
1000
500
0

Project BubbleIsland_Frontend_SNAPSHOT

Current feature development version of the frontend of Bubble Island. It is triggered by every change in the SCM. The appropriate files are deployed to the de

- Workspace
- Last Successful Artifacts
- Recent Changes
- Latest Test Result (no failures)

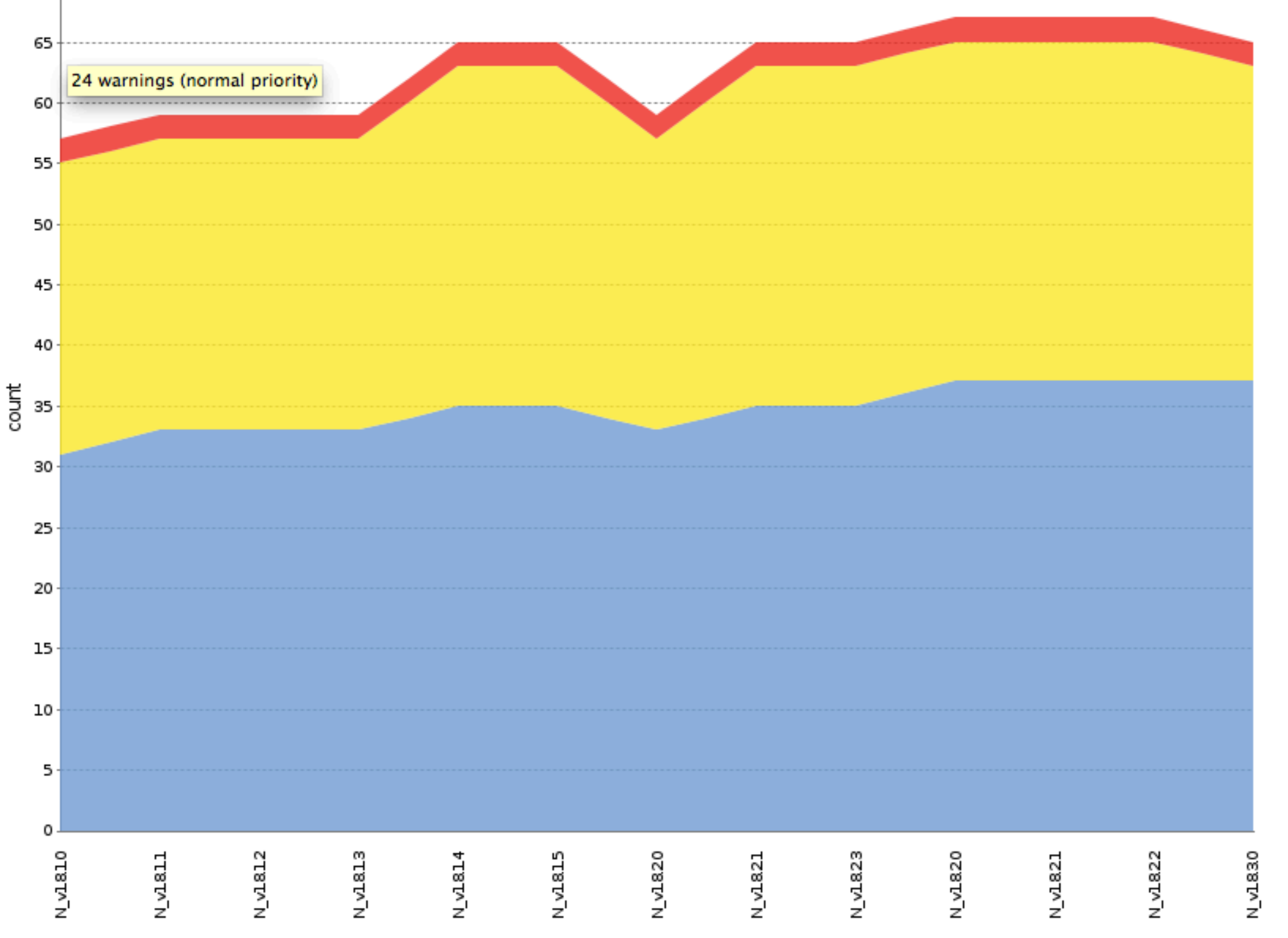
Build History (trend)

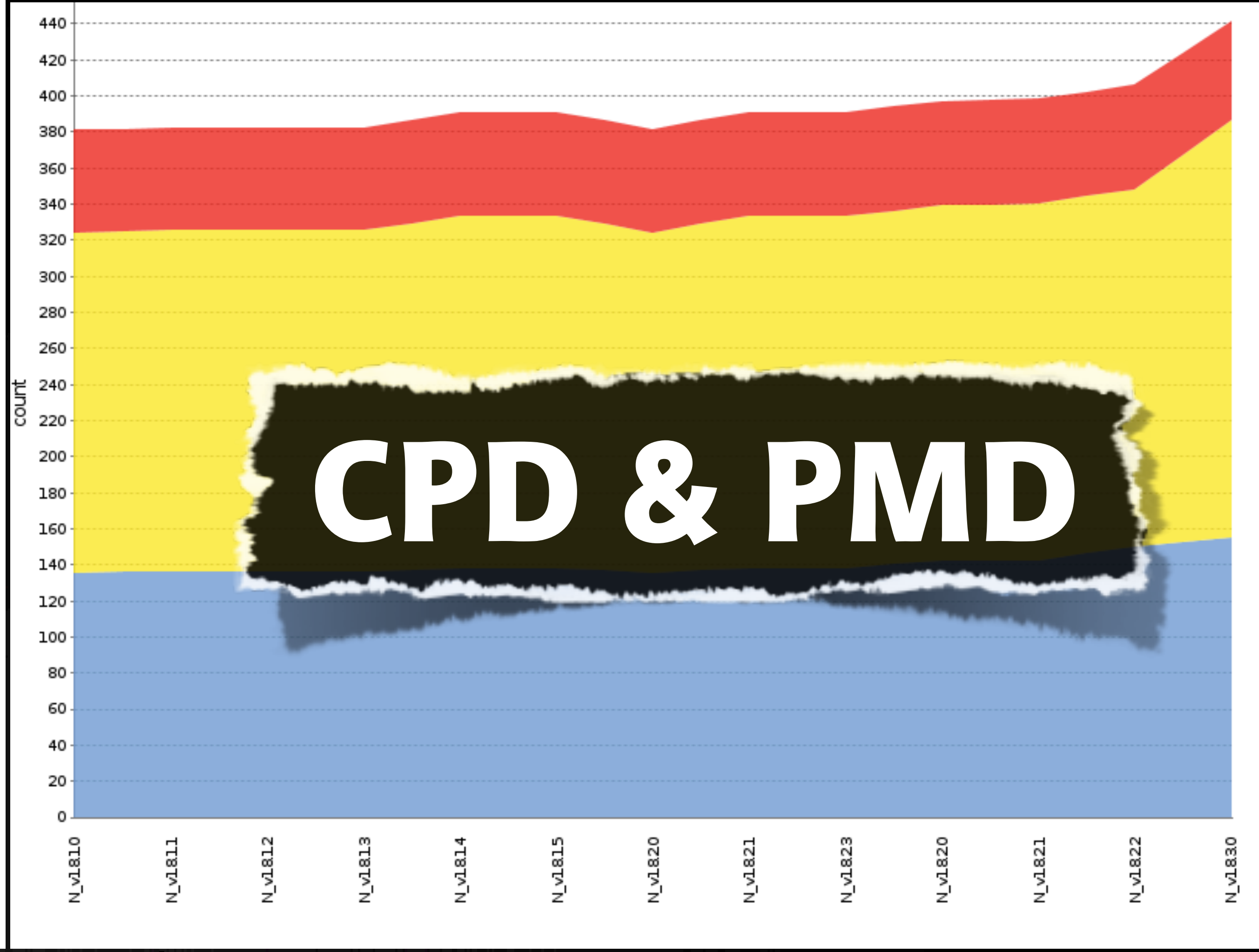
S_v1.8.3.12	09.08.2013 11:20:24	32MB
S_v1.8.3.11	09.08.2013 10:43:23	32MB
S_v1.8.3.10	09.08.2013 10:23:29	32MB
S_v1.8.3.9	08.08.2013 18:17:23	31MB
S_v1.8.3.8	08.08.2013 17:57:23	31MB
S_v1.8.3.7	08.08.2013 17:45:23	31MB
S_v1.8.3.6	08.08.2013 17:20:24	9MB
S_v1.8.3.5	08.08.2013 17:01:44	9MB
S_v1.8.3.4	08.08.2013 16:54:23	7MB
S_v1.8.3.3	08.08.2013 16:29:26	9MB
S_v1.8.3.2	08.08.2013 16:17:23	990KB
S_v1.8.3.1	08.08.2013 10:45:23	992KB
S_v1.8.3.0	08.08.2013 10:00:24	1MB
S_v1.8.2.24	08.08.2013 09:25:23	1MB
S_v1.8.2.23	30.07.2013 17:47:23	1MB
S_v1.8.2.22	30.07.2013 17:15:24	410KB
S_v1.8.2.21	30.07.2013 16:52:25	1MB
S_v1.8.2.20	30.07.2013 13:12:23	1MB
S_v1.8.2.19	30.07.2013 11:47:23	1MB
S_v1.8.2.18	29.07.2013 11:31:26	411KB
S_v1.8.2.17	25.07.2013 10:26:07	1MB

Permalinks

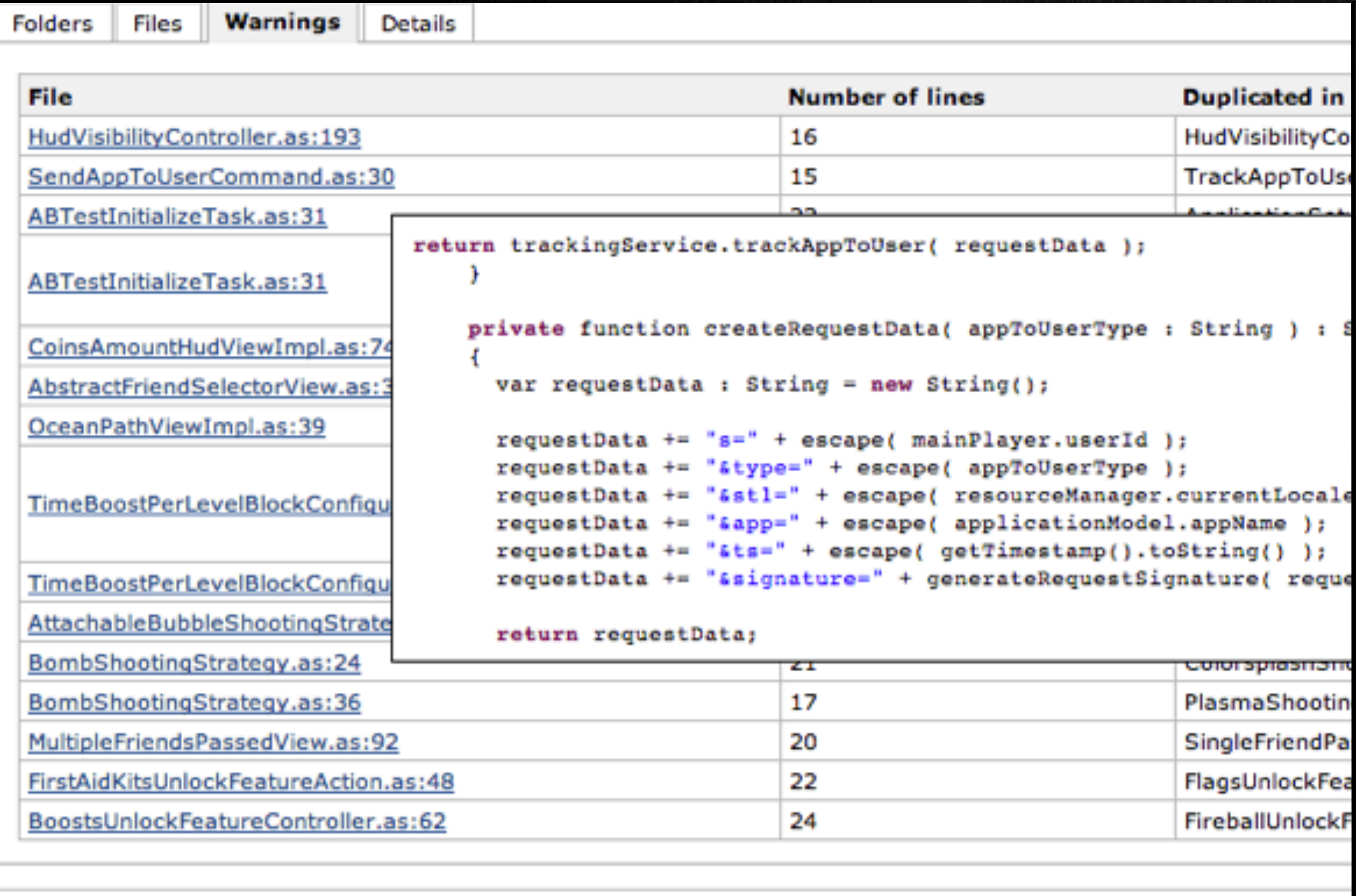
- Last build (S_v1.8.3.12), 2 days 4 hr ago
- Last stable build (S_v1.8.3.12), 2 days 4 hr ago
- Last successful build (S_v1.8.3.12), 2 days 4 hr ago
- Last failed build (S_v1.8.3.6), 2 days 22 hr ago
- Last unstable build (S_v1.8.0.15), 1 mo 4 days ago
- Last unsuccessful build (S_v1.8.3.6), 2 days 22 hr ago

http://jenkins.org



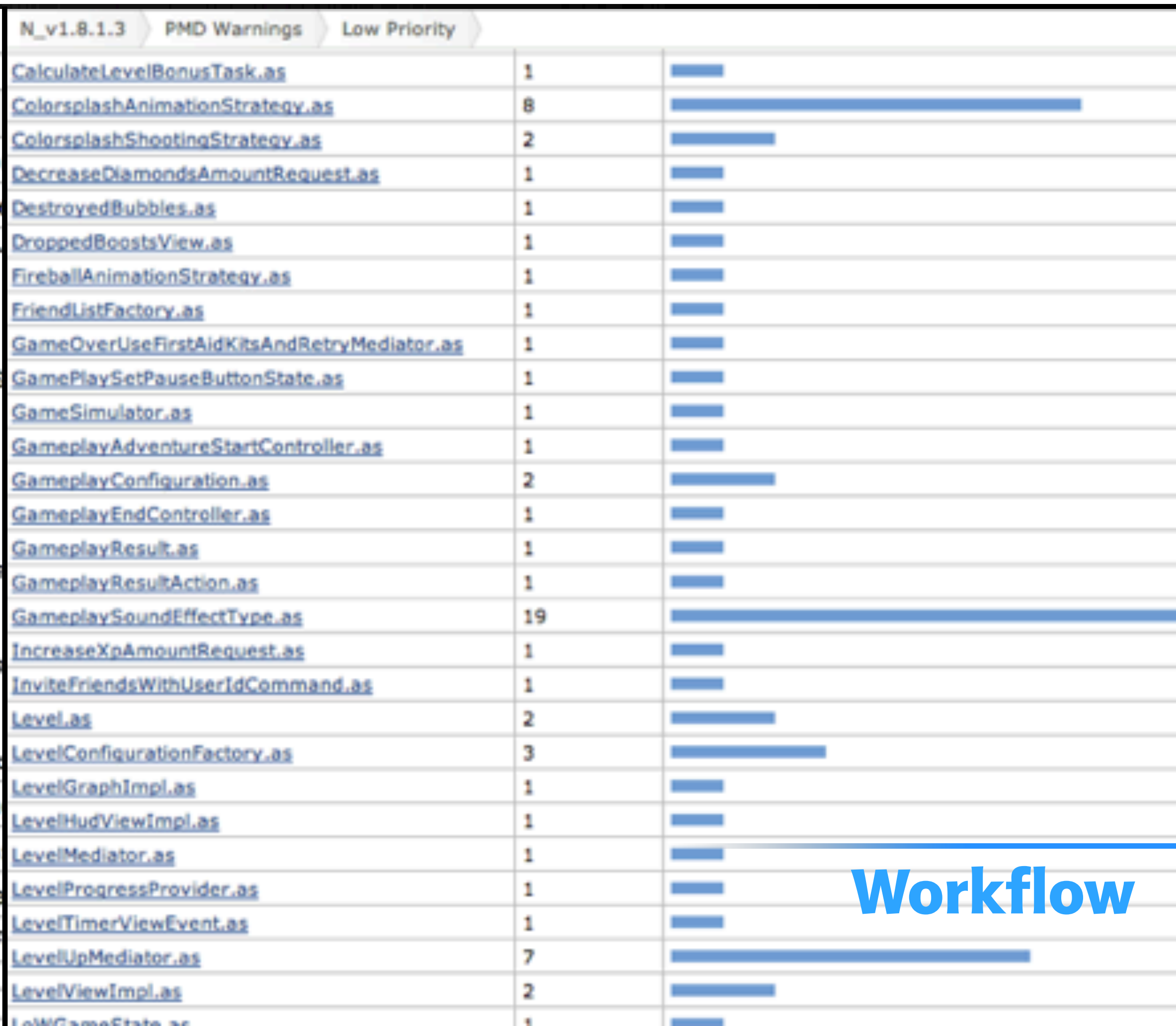


CPD & PMD



File	Number of lines	Duplicated in
HudVisibilityController.as:193	16	HudVisibilityCo
SendAppToUserCommand.as:30	15	TrackAppToUs
ABTestInitializeTask.as:31		
ABTestInitializeTask.as:31		
CoinsAmountHudViewImpl.as:7		
AbstractFriendSelectorView.as:		
OceanPathViewImpl.as:39		
TimeBoostPerLevelBlockConfigu		
TimeBoostPerLevelBlockConfigu		
AttachableBubbleShootingStrate		
BombShootingStrategy.as:24	21	ColorsplashSh
BombShootingStrategy.as:36	17	PlasmaShootin
MultipleFriendsPassedView.as:92	20	SingleFriendPa
FirstAidKitsUnlockFeatureAction.as:48	22	FlagsUnlockFea
BoostsUnlockFeatureController.as:62	24	FireballUnlockF

```
return trackingService.trackAppToUser( requestData );
    }

    private function createRequestData( appToUserType : String ) : S
    {
      var requestData : String = new String();

      requestData += "s=" + escape( mainPlayer.userId );
      requestData += "&type=" + escape( appToUserType );
      requestData += "&stl=" + escape( resourceManager.currentLocale
      requestData += "&app=" + escape( applicationModel.appName );
      requestData += "&ts=" + escape( getTimestamp().toString() );
      requestData += "&signature=" + generateRequestSignature( reque

      return requestData;
```



CalculateLevelBonusTask.as	1
ColorsplashAnimationStrategy.as	8
ColorsplashShootingStrategy.as	2
DecreaseDiamondsAmountRequest.as	1
DestroyedBubbles.as	1
DroppedBoostsView.as	1
FireballAnimationStrategy.as	1
FriendListFactory.as	1
GameOverUseFirstAidKitsAndRetryMediator.as	1
GamePlaySetPauseButtonState.as	1
GameSimulator.as	1
GameplayAdventureStartController.as	1
GameplayConfiguration.as	2
GameplayEndController.as	1
GameplayResult.as	1
GameplayResultAction.as	1
GameplaySoundEffectType.as	19
IncreaseXpAmountRequest.as	1
InviteFriendsWithUserIdCommand.as	1
Level.as	2
LevelConfigurationFactory.as	3
LevelGraphImpl.as	1
LevelHudViewImpl.as	1
LevelMediator.as	1
LevelProgressProvider.as	1
LevelTimerViewEvent.as	1
LevelUpMediator.as	7
LevelViewImpl.as	2

Workflow

Mehrwert für Wooga

Flexibles Teambuilding

http://www.flickr.com/photos/scoobay

Hohe Produktivität über langen Zeitraum

http://www.flickr.com/photos/stephenbegin

Schnelle Reaktion auf veränderte Anforderungen

75
RallyAMERICA
DREW
HIGGINS
BFGoodrich
SUBARU
STI
PUMA
MOTUL
Vermont SportsCar
PIAA
BFGoodrich
MOTUL
PIAA

http://www.flickr.com/photos/jimculp

Clean Code als gemeinsame Sprache

Weniger wtf/minute

Fragen?

jobs.wooga.com

@StephanPartzsch