# OTCv8 Dev Docs

# Start

# OTCv8 Dev — Dokument

a

c

j

a

Cel: kompletna baza wiedzy (build, architektura, API Lua/C++), z diagramami i przykładami.

* **Szybki start**: sekcja *Build*. - **Architektura**: diagramy Mermaid + graf modułów. - **API**: przykłady Lua/C++ (do rozszerzenia).

!!! info "Baza wiedzy dla edytora"

Twój edytor może konsumować `search/search\_index.json` z tej strony (GitHub Pages) jako indeks wiedzy...

# Architektura

# Architekt

u

r

a

(

s

k

r

ó

t

)

=== "Warstwy"

C++ Core – silnik render/UI/IO

Lua – logika modułów (vBot)

OTUI – deklaratywne layouty

=== "Kontrakty"

Eventy Lua ⟷ UI, IPC/WS, zasoby

# Moduły (vBot)

# Moduły (vBot) —

p

r

z

e

w

o

d

n

i

k

!!! info "Cel"

Jak tworzyć i ładować moduły Lua dla klienta OTCv8.

## Strukt

u

r

a

m

o

d

u

ł

u

## Minimalny m

o

d

u

ł

(

L

u

a

)

function M.start() print("my-module start") end  
  
function M.stop() print("my-module stop") end  
  
return M

## Rejestrowanie zdarzeń (

p

r

z

y

k

ł

a

d

)

## Kon

f

i

g

u

r

a

c

j

a

* config.lua – wartości domyślne (np. hotkeye, progi). - Pliki konfiguracyjne użytkownika trzymaj oddzielnie.

## D

e

b

u

g

/

l

o

g

i

* Pisz do konsoli lub pliku logs/my-module.log. - Dodaj flagę DEBUG=true i warunkowe logowanie.

## Dobr

e

p

r

a

k

t

y

k

i

* Nazwy przestrzeni modułu (my\_module.\*). - Brak efektów ubocznych przy require. - Komendy eksportuj jawnie (np. M.start, M.stop).

# Realtime (WebSocket)

# Realtime (

W

e

b

S

o

c

k

e

t

)

UI->>WS: handshake (JWT / token) WS->>S: connect S-->>UI: events: metrics, logs, char\_info UI->>S: cmd: START/STOP, settings

Z

a

s

a

d

y

* **WSS** + origin allowlist + rate-limit. - Autoryzacja w handshake (JWT / session). - Walidacja schematów wiadomości.

## Przykład (Node +

s

o

c

k

e

t

i

o

)

```ts io.use(authMiddleware); io.on("connection", (s) => { s.join(user:${s.user.id}); s.on("cmd", (payload) => { /\* validate + run \*/ }); });

---  
  
# OTUI  
  
# OTU  
  
I  
  
—  
  
p  
  
o  
  
d  
  
s  
  
t  
  
a  
  
w  
  
y  
  
!!! info  
  
 OTUI to deklaratywne layouty interfejsu użytkownika.  
  
## Przykł  
  
a  
  
d  
  
l  
  
a  
  
y  
  
o  
  
u  
  
t  
  
u  
  
```otui Panel id: main anchor: top left size: 400 300  
  
Label text: "Status: OK" anchors.centerIn: parent

## Zdarzenia

/

w

i

ą

z

a

n

i

a

* Właściwości elementów można powiązać z danymi (np. przez Lua). - Aktualizacje push przez eventy modułów.

W

s

k

a

z

ó

w

k

i

* Trzymaj layouty w layouts/\*. - Styluj wspólnymi klasami, nie inline.

# Windows

# Build — Windows

1. Zainstaluj **Visual Studio 2019** + **vcpkg**. 2. W vcpkg doinstaluj zależności (patrz README projektu). 3. Kompilacja wg instrukcji repo (tu wklej swoje komendy 1:1).

# Linux

# Build — Linux

1. Zależności systemowe (gcc/clang, cmake, itp.) 2. Kroki kompilacji.

# Android

# Build — Android

1. NDK/SDK, Java, skrypty assets. 2. Kroki kompilacji/apk.

# API (przykłady)

# Lua API (

p

r

z

y

k

ł

a

d

y

)

Docelowo: generowane referencje (LDoc/EmmyLua) + przykłady.

# Style guide

# Lua —

s

t

y

l

e

g

u

i

d

e

* snake\_case dla zmiennych i funkcji. - Moduły zwracają tabelę publicznego API. - Brak efektów ubocznych w require. - Pliki < 300 linii, funkcje < 50 linii. - Obsługa błędów: pcall/xpcall dla krytycznych ścieżek. - Logowanie warunkowe przez DEBUG flagę.

# Przegląd

# C+

—

p

r

z

e

g

l

ą

d

* Struktura katalogów - Kluczowe klasy/entrypoints - Jak rozszerzać (hooki, interfejsy)

# Narzędzia

N

a

r

z

ę

d

z

i

a

* Skrypty pomocnicze znajdziesz w tools/ repo. - Uzupełnij listę narzędzi po weryfikacji w repo.

# Contributing

# Co

n

t

r

i

b

u

t

i

n

g

1. Fork → branch feature/…. 2. Commity: konwencja type(scope): msg (np. docs(build): …). 3. PR z krótkim opisem i screenami. 4. Review: 1 LGTM + zielone CI.

S

t

y

l

e

* Lua: patrz *Lua → Style guide*. - C++: clang-format (domyślny styl projektu).

## Commity dok

u

m

e

n

t

a

c

j

i

* Zmiany w docs/\*\* nie uruchamiają CMake (paths-ignore).

# Testy

T

e

s

t

y

* Lua: testy jednostkowe (busted) — folder spec/. - C++: GoogleTest (jeśli dostępne). - E2E: scenariusze ręczne + checklisty w PR.

## Ra

p

o

r

t

b

ł

ę

d

u

* Kroki odtworzenia, logi, wersja, OS.

# Release

R

e

l

e

a

s

e

1. Bump wersji (semver) / tag. 2. Changelog z PR od ostatniego taga. 3. Build artefaktów (CI). 4. Publikacja + checksumy.

## Wersjonowanie dok

u

m

e

n

t

a

c

j

i

* Docsy budują się z gałęzi master.

# Security

S

e

c

u

r

i

t

y

* HTTPS/WSS, HSTS, twardy CSP i CORS. - Auth: krótkie JWT + refresh, RBAC. - WS: origin check, limit bufora, ping. - Walidacja payloadów (Zod/JSON Schema). - Sekrety nigdy w repo (env w CI).

Zgłoszenia luk: [security@twojadomena.example](mailto:security@twojadomena.example)

# Troubleshooting

# Troub

l

e

s

h

o

o

t

i

n

g

B

u

i

l

d

* Brak zależności vcpkg → zainstaluj pakiety i przebuduj cache.

A

n

d

r

o

i

d

* Błąd NDK: sprawdź wersję i ścieżki SDK/NDK.

W

S

* Rozłączenia: sprawdź pingTimeout/pingInterval oraz proxy.

# FAQ

F

A

Q

**Czy mogę używać swoich modułów?** Tak, wrzuć do modules/<nazwa> i zarejestruj w konfiguracji.

**Czy działa na Linux/Windows?** Tak — patrz sekcja *Build*.

# Słownik

S

ł

o

w

n

i

k

* **OTUI** — język layoutów UI. - **vBot** — moduły Lua automatyzujące zachowania. - **RAG** — Retrieval Augmented Generation (wyszukiwanie + LLM).

# Roadmap

R

o

a

d

m

a

p

* ☐ Pełna referencja API Lua (LDoc). - [ ] Generowanie Doxygen dla C++ i wpięcie do docs. - [ ] Więcej przykładów OTUI. - [ ] Testy E2E dashboardu.

# Pełne API (auto)

# OTCv8 – Pełne API (auto)

Wygenerowano: 2025-10-04T13:55:07.955Z

Ten plik jest generowany automatycznie z kodu. Nie edytuj ręcznie.

## 1. Lua

### 1.1. Zdarzenia (on\*) - onAddItem(...) - onAddThing(...) - onAddVip(...) - onAddonChange(...) - onAmountChange(...) - onAnimatedText(...) - onAppearanceChange(...) - onAttackingCreatureChange(...) - onAuraSelect(...) - onAutoWalkFail(...) - onBaseMagicLevelChange(...) - onBaseSkillChange(...) - onBaseSpeedChange(...) - onBattleButtonHoverChange(...) - onBattleButtonMouseRelease(...) - onBlessingsChange(...) - onBuyWithBackpackChange(...) - onCameraPositionChange(...) - onCancelWalk(...) - onChangeCategory(...) - onChangeOfferType(...) - onChangeSlotFilter(...) - onChangeSortOrder(...) - onChangeSortType(...) - onChangeSubCategory(...) - onChangeUseType(...) - onChannelEvent(...) - onChannelList(...) - onCharacterList(...) - onChildFocusChange(...) - onChooseItemMouseRelease(...) - onClick(...) - onClickIgnoreButton(...) - onClose(...) - onCloseChannel(...) - onCloseImbuementWindow(...) - onCloseNpcTrade(...) - onCoinBalance(...) - onColorCheckChange(...) - onColorModeChange(...) - onCommandChange(...) - onConfigChange(...) - onConnect(...) - onContainerChangeSize(...) - onContainerClose(...) - onContainerOpen(...) - onContainerUpdateItem(...) - onCreate(...) - onCreatureAppear(...) - onCreatureDisappear(...) - onCreatureHealthPercentChange(...) - onCreaturePositionChange(...) - onDecrement(...) - onDestroy(...) - onDoubleClick(...) - onDownload(...) - onDownloadProgress(...) - onDragEnter(...) - onDragLeave(...) - onDragMove(...) - onDrop(...) - onDropActionButton(...) - onEmblemChange(...) - onError(...) - onExperienceChange(...) - onExtendedJSONOpcode(...) - onExtendedOpcode(...) - onFilterSearch(...) - onFlagMouseRelease(...) - onFocusChange(...) - onFreeCapacityChange(...) - onGameCloseTrade(...) - onGameConnectionError(...) - onGameCounterTrade(...) - onGameEditList(...) - onGameEditText(...) - onGameEnd(...) - onGameLoginError(...) - onGameLoginToken(...) - onGameOwnTrade(...) - onGameQuestLine(...) - onGameQuestLog(...) - onGameStart(...) - onGameUpdateNeeded(...) - onGeometryChange(...) - onGet(...) - onGetProgress(...) - onGroupSpellCooldown(...) - onHTTPResult(...) - onHealthBarSelect(...) - onHealthChange(...) - onHeightChange(...) - onHotkeyTextChange(...) - onHover(...) - onHoverChange(...) - onIconChange(...) - onIgnoreCapacityChange(...) - onIgnoreEquippedChange(...) - onImbuementWindow(...) - onIncrement(...) - onInventoryChange(...) - onItemBoxChecked(...) - onItemChange(...) - onKeyDown(...) - onKeyPress(...) - onKeyUp(...) - onKeypadTouchMove(...) - onKeypadTouchPress(...) - onKeypadTouchRelease(...) - onLayoutUpdate(...) - onLevelChange(...) - onLoadCallback(...) - onLocaleChanged(...) - onLog(...) - onLoginAdvice(...) - onLoginError(...) - onLoginWait(...) - onLogout(...) - onMagicLevelChange(...) - onManaBarSelect(...) - onManaChange(...) - onMarketBrowse(...) - onMarketDetail(...) - onMarketEnter(...) - onMarketLeave(...) - onMarketMessage(...) - onMessage(...) - onMiniWindowClose(...) - onMissle(...) - onModalDialog(...) - onMountButtonClick(...) - onMountSelect(...) - onMouseGrabberRelease(...) - onMouseMove(...) - onMousePress(...) - onMouseRelease(...) - onMouseWheel(...) - onMovementChange(...) - onOfflineTrainingChange(...) - onOpcode(...) - onOpen(...) - onOpenChannel(...) - onOpenNpcTrade(...) - onOpenOwnPrivateChannel(...) - onOpenPrivateChannel(...) - onOpenPvpSituationsChange(...) - onOptionChange(...) - onOutfitChange(...) - onOutfitSelect(...) - onOverlayGeometryChange(...) - onPiecePriceChange(...) - onPlayerGoods(...) - onPlayerHealthChange(...) - onPlayerInventoryChange(...) - onPlayerPositionChange(...) - onPositionChange(...) - onPost(...) - onPostProgress(...) - onPresetSelect(...) - onPreyActive(...) - onPreyFreeRolls(...) - onPreyInactive(...) - onPreyLocked(...) - onPreyPrice(...) - onPreySelection(...) - onPreyTimeLeft(...) - onProfileChange(...) - onProtocolError(...) - onProxyList(...) - onQuantityValueChange(...) - onRecv(...) - onRegenerationChange(...) - onRemoveItem(...) - onRemoveThing(...) - onResourceBalance(...) - onRootGeometryUpdate(...) - onRuleViolationCancel(...) - onRuleViolationChannel(...) - onRuleViolationLock(...) - onRuleViolationRemove(...) - onSave(...) - onScrollHeightChange(...) - onScrollWidthChange(...) - onSearchTextChange(...) - onSelectBuyOffer(...) - onSelectHotkeyLabel(...) - onSelectMyBuyOffer(...) - onSelectMySellOffer(...) - onSelectReason(...) - onSelectSellOffer(...) - onSendAutomaticallyChange(...) - onServerChange(...) - onSessionKey(...) - onSetChaseMode(...) - onSetFightMode(...) - onSetPVPMode(...) - onSetSafeFight(...) - onSetSafeFight2(...) - onSetup(...) - onShaderSelect(...) - onShieldChange(...) - onShowAllItemsChange(...) - onShowAuraChange(...) - onShowBarsChange(...) - onShowFloorChange(...) - onShowMountChange(...) - onShowOutfitChange(...) - onShowShaderChange(...) - onShowWingsChange(...) - onSkillButtonClick(...) - onSkillChange(...) - onSkullChange(...) - onSoulChange(...) - onSpeedChange(...) - onSpellCooldown(...) - onSpellGroupCooldown(...) - onStaminaChange(...) - onStatesChange(...) - onStaticText(...) - onStoreCategories(...) - onStoreError(...) - onStoreInit(...) - onStoreOffers(...) - onStorePurchase(...) - onStoreTransactionHistory(...) - onStyleApply(...) - onTabChange(...) - onTabClick(...) - onTabDragEnter(...) - onTabDragLeave(...) - onTabDragMove(...) - onTabMousePress(...) - onTabMouseRelease(...) - onTalk(...) - onTeleport(...) - onTextAreaUpdate(...) - onTextChange(...) - onTextMessage(...) - onTibia12HTTPResult(...) - onTotalCapacityChange(...) - onTotalPriceChange(...) - onTouchRelease(...) - onTrackOptionChange(...) - onTradeClick(...) - onTradeTypeChange(...) - onTradeWith(...) - onTurn(...) - onTypeChange(...) - onUnjustifiedPointsChange(...) - onUpdateNeeded(...) - onUse(...) - onUseWith(...) - onValueChange(...) - onVipListLabelMousePress(...) - onVipListMousePress(...) - onVipStateChange(...) - onVisibilityChange(...) - onWalk(...) - onWalkFinish(...) - onWidgetHoverChange(...) - onWidgetKeyDown(...) - onWidgetKeyPress(...) - onWidgetKeyUp(...) - onWidgetStyleApply(...) - onWingsSelect(...) - onWsClose(...) - onWsError(...) - onWsMessage(...) - onWsOpen(...) - onZoomChange(...)

### 1.2. Kontekst ctx.\* *brak*

### 1.3. Moduły (eksporty M.\*) *brak*

### 1.4. Globalne funkcje (heur.) - abort() - about\_graphics() - about\_modules() - about\_version() - abs() - accept() - action() - add() - alarm() - api() - assert() - atan2() - attack() - back() - band() - bit() - bonus() - booleantonumber() - button() - buy() - bxor() - byte() - callback() - cancel() - cast() - cavebot() - ceil() - center() - challenge() - changed() - char() - check() - chodzenie() - clean() - clear() - clearbit() - client() - close() - codepoint\_to\_utf8() - collect() - colortostring() - comma\_value() - compare() - concat() - condition() - config() - connect() - container() - contains() - copy() - cos() - countbrackets() - create() - create\_set() - date() - debuff() - decode() - decode\_error() - decrement() - decrypt() - delay() - destory() - destroy() - difftime() - direction() - directory() - dirtostring() - disable() - disconnect() - dismount() - display() - distance() - dofile() - dofiles() - down() - download() - draw\_debug\_boxes() - dump() - edit() - editor() - elseif() - empty() - enable() - encode() - encode\_nil() - encode\_number() - encode\_string() - encode\_table() - encrypt() - ends() - enqueue() - eof() - equal() - equals() - error() - escape\_char() - exist() - exists() - exit() - exp() - explode() - export() - extends() - extension() - extract() - fail() - fatal() - file() - files() - fill() - find() - findbyfield() - findbykey() - findkey() - flash() - float() - floor() - fmod() - focus() - follow() - format() - format\_thousand() - freecap() - frexp() - fromboolean() - func() - gcinfo() - get() - getfenv() - getfsrcpath() - getinfo() - getmetatable() - getn() - getname() - gmatch() - gsub() - haskey() - helper() - hide() - hide\_map() - high() - hotkey() - hppercent() - import() - increment() - info() - information() - init() - insert() - invalid() - inventory() - ipairs() - iptostring() - isinteger() - issues() - isu16() - isu32() - isu64() - isu8() - ldexp() - left() - len() - level() - line() - lines() - list() - listen() - load() - loadasmodule() - loadfile() - loadstring() - lock() - log() - login() - logs() - look() - lose() - lower() - lvl() - macro() - macros() - make\_indent() - makedecoder() - makeencoder() - makesingleton() - mana() - manapercent() - match() - matchcount() - max() - maximize() - menu() - merge() - message() - micros() - millis() - min() - minimize() - mod() - module\_loader() - monitor() - mount() - move() - music() - name() - new() - newclass() - next() - next\_char() - nfiles() - number() - numbertoboolean() - off() - offline() - opcode() - open() - order() - pack() - pairs() - params() - parse() - parse\_array() - parse\_literal() - parse\_number() - parse\_object() - parse\_string() - parse\_unicode\_escape() - pathfinder() - pcall() - pcolored() - pdebug() - permute() - perror() - pinfo() - ping() - place() - play() - pointtostring() - popvalue() - pos() - position() - post() - postostring() - posx() - posy() - posz() - pow() - preload() - prev() - process() - proper() - properly() - protectedcall() - pwarning() - quit() - raise() - random() - randomness() - randomseed() - range() - rawget() - recttostring() - recursivecopy() - recv() - refresh() - reload() - remove() - removevalue() - rep() - report() - reset() - resize() - resolvepath() - restart() - retries() - reverse() - rotate() - round() - rpairs() - run() - runinsandbox() - sandboxed() - save() - say() - scale() - schedule() - script() - second() - seconds() - see() - select() - selectivecopy() - send() - separatly() - server() - servidor() - serwera() - set() - setbit() - setfenv() - setmetatable() - setter() - setup() - shl() - show() - show\_map() - shr() - signalcall() - silent() - sin() - singlehotkey() - size() - sizetostring() - slow() - sort() - split() - splitlines() - sqrt() - stamina() - starts() - startup() - stop() - storage() - sub() - summons() - table() - talk() - target() - terminate() - test() - text() - the() - time() - toboolean() - tocolor() - toggle() - tonumber() - topoint() - torect() - tosize() - tostring() - traceback() - trim() - turn() - type() - types() - underscore() - unexport() - unload() - unlock() - unpack() - update() - upper() - use() - usewith() - validate() - value() - values() - voc() - void() - wait() - walk() - warn() - warning() - wrap()

## 2. OTUI (layouty) ### layouts/mobile/styles/10-scrollbars.otui - sliderButton — **ScrollBarSlider** - valueLabel — **ScrollBarValueLabel** - decrementButton — **UIButton** - incrementButton — **UIButton** - decrementButton — **UIButton** - incrementButton — **UIButton**

### layouts/mobile/styles/20-smallscrollbar.otui - decrementButton — **UIButton** - incrementButton — **UIButton** - sliderButton — **UIButton** - valueLabel — **Label**

### layouts/mobile/styles/30-miniwindow.otui - miniwindowTopBar — **UIWidget** - closeButton — **UIButton** - minimizeButton — **UIButton** - lockButton — **UIButton** - miniwindowScrollBar — **VerticalScrollBar** - bottomResizeBorder — **ResizeBorder** - contentsPanel — **MiniWindowContents**

### layouts/mobile/styles/40-console.otui - consoleTab — **ConsoleTabBarPanel** - consoleBuffer — **ScrollablePanel** - consoleScrollBar — **VerticalScrollBar** - toggleChat — **CheckBox** - prevChannelButton — **TabButton** - consoleTabBar — **ConsoleTabBar** - nextChannelButton — **TabButton** - closeChannelButton — **TabButton** - clearChannelButton — **TabButton** - channelsButton — **TabButton** - ignoreButton — **TabButton** - consoleContentPanel — **Panel** - sayModeButton — **TabButton** - consoleTextEdit — **TextEdit**

### layouts/mobile/styles/40-inventory.otui - slot1 — **HeadSlot** - slot4 — **BodySlot** - slot7 — **LegSlot** - slot8 — **FeetSlot** - slot2 — **NeckSlot** - slot6 — **LeftSlot** - slot9 — **FingerSlot** - slot3 — **BackSlot** - slot5 — **RightSlot** - slot10 — **AmmoSlot** - purseButton — **PurseButton** - inventoryWindow — **InventoryWindow** - inventoryPanel — **Panel** - soulLabel — **SoulCapLabel** - capLabel — **SoulCapLabel** - conditionPanel — **Panel** - fightOffensiveBox — **FightOffensiveBox** - chaseModeBox — **ChaseModeBox** - fightBalancedBox — **FightBalancedBox** - safeFightBox — **SafeFightBox** - fightDefensiveBox — **FightDefensiveBox** - mountButton — **MountButton** - buttonsPanel — **Panel** - buttonPvp — **UIButton**

### layouts/retro/styles/20-tabbars.otui - buttonsPanel — **TabBar** - buttonsPanel — **TabBarVertical** - scrollBar — **VerticalScrollBar**

### layouts/retro/styles/20-topmenu.otui - topMenu — **TopMenu** - discord — **UIWidget** - discordLabel — **Label** - rightButtonsPanel — **TopMenuButtonsPanel** - rightGameButtonsPanel — **TopMenuButtonsPanel** - onlineLabel — **Label** - leftButtonsPanel — **TopMenuButtonsPanel** - leftGameButtonsPanel — **TopMenuButtonsPanel**

### layouts/retro/styles/30-miniwindow.otui - miniwindowTopBar — **UIWidget** - closeButton — **UIButton** - minimizeButton — **UIButton** - lockButton — **UIButton** - miniwindowScrollBar — **VerticalScrollBar** - bottomResizeBorder — **ResizeBorder** - contentsPanel — **MiniWindowContents** - minimizeButton — **UIButton** - miniwindowTopBar — **UIWidget** - closeButton — **UIButton** - miniwindowScrollBar — **VerticalScrollBar** - bottomResizeBorder — **ResizeBorder**

### layouts/retro/styles/40-console.otui - consoleTab — **ConsoleTabBarPanel** - consoleBuffer — **ScrollablePanel** - consoleScrollBar — **VerticalScrollBar** - toggleChat — **CheckBox** - prevChannelButton — **TabButton** - consoleTabBar — **ConsoleTabBar** - nextChannelButton — **TabButton** - closeChannelButton — **TabButton** - clearChannelButton — **TabButton** - channelsButton — **TabButton** - ignoreButton — **TabButton** - consoleContentPanel — **Panel** - sayModeButton — **TabButton** - separator — **HorizontalSeparator** - consoleTextEdit — **TextEdit**

### layouts/retro/styles/40-gamebuttons.otui - buttons — **Panel**

### layouts/retro/styles/40-healthinfo.otui - experienceBar — **ExperienceBar** - soulLabel — **SoulLabel** - capLabel — **CapLabel** - healthOverlay — **HealthOverlay** - topHealthBar — **HealthBar** - topManaBar — **ManaBar** - healthCircle — **UIProgressBar** - healthCircleFront — **UIProgressBar** - manaCircle — **UIProgressBar** - manaCircleFront — **UIProgressBar** - healthBar — **HealthBar** - manaBar — **ManaBar** - conditionPanel — **ExperienceBar**

### layouts/retro/styles/40-inventory.otui - slot1 — **HeadSlot** - slot4 — **BodySlot** - slot7 — **LegSlot** - slot8 — **FeetSlot** - slot2 — **NeckSlot** - slot6 — **LeftSlot** - slot9 — **FingerSlot** - slot3 — **BackSlot** - slot5 — **RightSlot** - slot10 — **AmmoSlot** - purseButton — **PurseButton** - inventoryWindow — **InventoryWindow** - minimizeButton — **UIButton** - inventoryPanel — **Panel** - soulLabel — **SoulCapLabel** - capLabel — **SoulCapLabel** - conditionPanel — **Panel** - fightOffensiveBox — **FightOffensiveBox** - chaseModeBox — **ChaseModeBox** - fightBalancedBox — **FightBalancedBox** - safeFightBox — **SafeFightBox** - fightDefensiveBox — **FightDefensiveBox** - mountButton — **MountButton** - buttonsPanel — **Panel** - buttonPvp — **UIButton**

### layouts/retro/styles/40-minimap.otui - floorUpWidget — **MinimapFloorUpButton** - floorDownWidget — **MinimapFloorDownButton** - zoomInWidget — **MinimapZoomInButton** - zoomOutWidget — **MinimapZoomOutButton** - resetWidget — **MinimapResetButton** - position — **Label** - description — **TextEdit** - flag0 — **MinimapFlagCheckBox** - flag1 — **MinimapFlagCheckBox** - flag2 — **MinimapFlagCheckBox** - flag3 — **MinimapFlagCheckBox** - flag4 — **MinimapFlagCheckBox** - flag5 — **MinimapFlagCheckBox** - flag6 — **MinimapFlagCheckBox** - flag7 — **MinimapFlagCheckBox** - flag8 — **MinimapFlagCheckBox** - flag9 — **MinimapFlagCheckBox** - flag10 — **MinimapFlagCheckBox** - flag11 — **MinimapFlagCheckBox** - flag12 — **MinimapFlagCheckBox** - flag13 — **MinimapFlagCheckBox** - flag14 — **MinimapFlagCheckBox** - flag15 — **MinimapFlagCheckBox** - flag16 — **MinimapFlagCheckBox** - flag17 — **MinimapFlagCheckBox** - flag18 — **MinimapFlagCheckBox** - flag19 — **MinimapFlagCheckBox** - okButton — **Button** - cancelButton — **Button** - minimap — **Minimap**

### layouts/retro/styles/40-outfitwindow.otui - creature — **UICreature** - title — **Label** - rename — **Panel** - input — **TextEdit** - save — **Button** - outfit — **UICreature** - bar — **Panel** - name — **Label** - preview — **MiniPanel** - options — **Panel** - showFloor — **FlatPanel** - check — **CheckBox** - showOutfit — **FlatPanel** - check — **CheckBox** - showMount — **FlatPanel** - check — **CheckBox** - showWings — **FlatPanel** - check — **CheckBox** - showAura — **FlatPanel** - check — **CheckBox** - showShader — **FlatPanel** - check — **CheckBox** - showBars — **FlatPanel** - check — **CheckBox** - panel — **FlatPanel** - floor — **Panel** - creature — **UICreature** - bars — **Panel** - name — **Label** - healthBar — **Panel** - image — **Panel** - manaBar — **Panel** - image — **Panel** - movement — **ChaseModeBox** - configure — **MiniPanel** - addon1 — **FlatPanel** - check — **CheckBox** - addon2 — **FlatPanel** - check — **CheckBox** - mount — **FlatPanel** - check — **CheckBox** - appearance — **MiniPanel** - settings — **ScrollablePanel** - preset — **Panel** - check — **CheckBox** - name — **FlatPanel** - outfit — **Panel** - check — **CheckBox** - name — **FlatPanel** - mount — **Panel** - check — **CheckBox** - name — **FlatPanel** - wings — **Panel** - check — **CheckBox** - name — **FlatPanel** - aura — **Panel** - check — **CheckBox** - name — **FlatPanel** - shader — **Panel** - check — **CheckBox** - name — **FlatPanel** - healthBar — **Panel** - check — **CheckBox** - name — **FlatPanel** - manaBar — **Panel** - check — **CheckBox** - name — **FlatPanel** - scrollbar — **VerticalScrollBar** - colorMode — **Panel** - head — **ButtonBox** - colorBoxPanel — **Panel** - presetButtons — **MiniPanel** - listSearch — **MiniPanel** - search — **TextEdit** - presetsList — **ScrollablePanel** - presetsScroll — **VerticalScrollBar** - selectionList — **ScrollablePanel** - selectionScroll — **VerticalScrollBar** - separator — **HorizontalSeparator**

## 3. WebSocket ### 3.1. Typy wiadomości (wykryte) - MIT - OpenCollective - array - boolean - commonjs - git - github - integer - module - object - opencollective - string - time-permitting - value

### 3.2. Schematy JSON ($id → plik) - https://raw.githubusercontent.com/DavidAnson/markdownlint/v0.38.0/schema/markdownlint-config-schema-strict.json → node\_modules/markdownlint/schema/markdownlint-config-schema-strict.json - https://raw.githubusercontent.com/DavidAnson/markdownlint/v0.38.0/schema/markdownlint-config-schema.json → node\_modules/markdownlint/schema/markdownlint-config-schema.json

## 4. C++ (nagłówki) - **src/framework/global.h**: extern void fatalError(const char\* error, const char\* file, int line) - **src/android/android\_native\_app\_glue.h**: int8\_t android\_app\_read\_cmd(struct android\_app\* android\_app) — Call when ALooper\_pollAll() returns LOOPER\_ID\_MAIN, reading the next app command message. - **src/android/android\_native\_app\_glue.h**: void android\_app\_pre\_exec\_cmd(struct android\_app\* android\_app, int8\_t cmd) — Call with the command returned by android\_app\_read\_cmd() to do the initial pre-processing of the given command. You can perform your own actions for the command after calling this function. - **src/android/android\_native\_app\_glue.h**: void android\_app\_post\_exec\_cmd(struct android\_app\* android\_app, int8\_t cmd) — Call with the command returned by android\_app\_read\_cmd() to do the final post-processing of the given command. You must have done your own actions for the command before calling this function. - **src/android/android\_native\_app\_glue.h**: extern void android\_main(struct android\_app\* app) — This is the function that application code must implement, representing the main entry to the app. - **src/client/walkmatrix.h**: return clear() - **src/client/uisprite.h**: void drawSelf(Fw::DrawPane drawPane) - **src/client/uisprite.h**: void setSpriteId(uint32 id) - **src/client/uisprite.h**: void onStyleApply(const std::string& styleName, const OTMLNodePtr& styleNode) - **src/client/uiprogressrect.h**: void drawSelf(Fw::DrawPane drawPane) - **src/client/uiprogressrect.h**: void setPercent(float percent) - **src/client/uiprogressrect.h**: void onStyleApply(const std::string& styleName, const OTMLNodePtr& styleNode) - **src/client/uiminimap.h**: void drawSelf(Fw::DrawPane drawPane) - **src/client/uiminimap.h**: bool setZoom(int zoom) - **src/client/uiminimap.h**: void setCameraPosition(const Position& pos) - **src/client/uiminimap.h**: bool floorUp() - **src/client/uiminimap.h**: bool floorDown() - **src/client/uiminimap.h**: Point getTilePoint(const Position& pos) - **src/client/uiminimap.h**: Rect getTileRect(const Position& pos) - **src/client/uiminimap.h**: Position getTilePosition(const Point& mousePos) - **src/client/uiminimap.h**: void anchorPosition(const UIWidgetPtr& anchoredWidget, Fw::AnchorEdge anchoredEdge, const Position& hookedPosition, Fw::AnchorEdge hookedEdge) - **src/client/uiminimap.h**: void fillPosition(const UIWidgetPtr& anchoredWidget, const Position& hookedPosition) - **src/client/uiminimap.h**: void centerInPosition(const UIWidgetPtr& anchoredWidget, const Position& hookedPosition) - **src/client/uiminimap.h**: virtual void onZoomChange(int zoom, int oldZoom) - **src/client/uiminimap.h**: virtual void onCameraPositionChange(const Position& position, const Position& oldPosition) - **src/client/uiminimap.h**: virtual void onStyleApply(const std::string& styleName, const OTMLNodePtr& styleNode) - **src/client/uiminimap.h**: void update() - **src/client/uimapanchorlayout.h**: int getHookedPoint(const UIWidgetPtr& hookedWidget, const UIWidgetPtr& parentWidget) - **src/client/uimapanchorlayout.h**: void centerInPosition(const UIWidgetPtr& anchoredWidget, const Position& hookedPosition) - **src/client/uimapanchorlayout.h**: void fillPosition(const UIWidgetPtr& anchoredWidget, const Position& hookedPosition) - **src/client/uimap.h**: bool onMouseMove(const Point& mousePos, const Point& mouseMoved) - **src/client/uimap.h**: void drawSelf(Fw::DrawPane drawPane) - **src/client/uimap.h**: void movePixels(int x, int y) - **src/client/uimap.h**: bool setZoom(int zoom) - **src/client/uimap.h**: bool zoomIn() - **src/client/uimap.h**: bool zoomOut() - **src/client/uimap.h**: void setVisibleDimension(const Size& visibleDimension) - **src/client/uimap.h**: void setKeepAspectRatio(bool enable) - **src/client/uimap.h**: Position getPosition(const Point& mousePos) - **src/client/uimap.h**: Point getPositionOffset(const Point& mousePos) - **src/client/uimap.h**: TilePtr getTile(const Point& mousePos) - **src/client/uimap.h**: virtual void onStyleApply(const std::string& styleName, const OTMLNodePtr& styleNode) - **src/client/uimap.h**: virtual void onGeometryChange(const Rect& oldRect, const Rect& newRect) - **src/client/uimap.h**: void updateVisibleDimension() - **src/client/uimap.h**: void updateMapSize() - **src/client/uiitem.h**: void drawSelf(Fw::DrawPane drawPane) - **src/client/uiitem.h**: void setItemId(int id) - **src/client/uiitem.h**: void setItemCount(int count) - **src/client/uiitem.h**: void setItemSubType(int subType) - **src/client/uiitem.h**: void setItem(const ItemPtr& item) - **src/client/uiitem.h**: void setItemShader(const std::string& str) - **src/client/uiitem.h**: void onStyleApply(const std::string& styleName, const OTMLNodePtr& styleNode) - **src/client/uiitem.h**: void cacheCountText() - **src/client/uigraph.h**: void drawSelf(Fw::DrawPane drawPane) - **src/client/uigraph.h**: void clear() - **src/client/uigraph.h**: void addValue(int value, bool ignoreSmallValues = false) - **src/client/uigraph.h**: void onStyleApply(const std::string& styleName, const OTMLNodePtr& styleNode) - **src/client/uicreature.h**: void drawSelf(Fw::DrawPane drawPane) - **src/client/uicreature.h**: void setOutfit(const Outfit& outfit) - **src/client/uicreature.h**: void setCenter(bool value) - **src/client/uicreature.h**: void onStyleApply(const std::string& styleName, const OTMLNodePtr& styleNode) - **src/client/towns.h**: void addTown(const TownPtr& town) - **src/client/towns.h**: void removeTown(uint32 townId) - **src/client/towns.h**: const TownPtr& getTown(uint32 townId) - **src/client/towns.h**: const TownPtr& getTownByName(std::string name) - **src/client/towns.h**: void sort() - **src/client/towns.h**: TownList::iterator findTown(uint32 townId) - **src/client/tile.h**: void calculateCorpseCorrection() - **src/client/tile.h**: void drawGround(const Point& dest, LightView\* lightView = nullptr) - **src/client/tile.h**: void drawBottom(const Point& dest, LightView\* lightView = nullptr) - **src/client/tile.h**: void drawCreatures(const Point& dest, LightView\* lightView = nullptr) - **src/client/tile.h**: void drawTop(const Point& dest, LightView\* lightView = nullptr) - **src/client/tile.h**: void drawTexts(Point dest) - **src/client/tile.h**: void drawWidget(Point dest) - **src/client/tile.h**: void clean() - **src/client/tile.h**: void addWalkingCreature(const CreaturePtr& creature) - **src/client/tile.h**: void removeWalkingCreature(const CreaturePtr& creature) - **src/client/tile.h**: void addThing(const ThingPtr& thing, int stackPos) - **src/client/tile.h**: bool removeThing(ThingPtr thing) - **src/client/tile.h**: ThingPtr getThing(int stackPos) - **src/client/tile.h**: EffectPtr getEffect(uint16 id) - **src/client/tile.h**: bool hasThing(const ThingPtr& thing) - **src/client/tile.h**: int getThingStackPos(const ThingPtr& thing) - **src/client/tile.h**: ThingPtr getTopThing() - **src/client/tile.h**: ThingPtr getTopLookThing() - **src/client/tile.h**: ThingPtr getTopLookThingEx(Point offset) - **src/client/tile.h**: ThingPtr getTopUseThing() - **src/client/tile.h**: CreaturePtr getTopCreature() - **src/client/tile.h**: CreaturePtr getTopCreatureEx(Point offset) - **src/client/tile.h**: ThingPtr getTopMoveThing() - **src/client/tile.h**: ThingPtr getTopMultiUseThing() - **src/client/tile.h**: ThingPtr getTopMultiUseThingEx(Point offset) - **src/client/tile.h**: std::vector<ItemPtr> getItems() - **src/client/tile.h**: std::vector<CreaturePtr> getCreatures() - **src/client/tile.h**: ItemPtr getGround() - **src/client/tile.h**: int getGroundSpeed() - **src/client/tile.h**: uint8 getMinimapColorByte() - **src/client/tile.h**: bool isPathable() - **src/client/tile.h**: bool isWalkable(bool ignoreCreatures = false) - **src/client/tile.h**: bool isFullGround() - **src/client/tile.h**: bool isFullyOpaque() - **src/client/tile.h**: bool isSingleDimension() - **src/client/tile.h**: bool isLookPossible() - **src/client/tile.h**: bool isBlockingProjectile() - **src/client/tile.h**: bool isClickable() - **src/client/tile.h**: bool isEmpty() - **src/client/tile.h**: bool isDrawable() - **src/client/tile.h**: bool mustHookSouth() - **src/client/tile.h**: bool mustHookEast() - **src/client/tile.h**: bool hasCreature() - **src/client/tile.h**: bool hasBlockingCreature() - **src/client/tile.h**: bool limitsFloorsView(bool isFreeView = false) - **src/client/tile.h**: bool canErase() - **src/client/tile.h**: int getElevation() - **src/client/tile.h**: bool hasElevation(int elevation = 1) - **src/client/tile.h**: void setText(const std::string& text, Color color) - **src/client/tile.h**: std::string getText() - **src/client/tile.h**: void setTimer(int time, Color color) - **src/client/tile.h**: int getTimer() - **src/client/tile.h**: void setFill(Color color) - **src/client/tile.h**: bool canShoot(int distance) - **src/client/tile.h**: void checkTranslucentLight() - **src/client/thingtypemanager.h**: void init() - **src/client/thingtypemanager.h**: void terminate() - **src/client/thingtypemanager.h**: void check() - **src/client/thingtypemanager.h**: bool loadDat(std::string file) - **src/client/thingtypemanager.h**: bool loadOtml(std::string file) - **src/client/thingtypemanager.h**: void loadOtb(const std::string& file) - **src/client/thingtypemanager.h**: void loadXml(const std::string& file) - **src/client/thingtypemanager.h**: void parseItemType(uint16 id, TiXmlElement \*elem) - **src/client/thingtypemanager.h**: void saveDat(std::string fileName) - **src/client/thingtypemanager.h**: void dumpTextures(std::string dir) - **src/client/thingtypemanager.h**: void replaceTextures(std::string dir) - **src/client/thingtypemanager.h**: void addItemType(const ItemTypePtr& itemType) - **src/client/thingtypemanager.h**: const ItemTypePtr& findItemTypeByClientId(uint16 id) - **src/client/thingtypemanager.h**: const ItemTypePtr& findItemTypeByName(std::string name) - **src/client/thingtypemanager.h**: ItemTypeList findItemTypesByName(std::string name) - **src/client/thingtypemanager.h**: ItemTypeList findItemTypesByString(std::string str) - **src/client/thingtypemanager.h**: const ThingTypePtr& getThingType(uint16 id, ThingCategory category) - **src/client/thingtypemanager.h**: const ItemTypePtr& getItemType(uint16 id) - **src/client/thingtypemanager.h**: ThingTypeList findThingTypeByAttr(ThingAttr attr, ThingCategory category) - **src/client/thingtypemanager.h**: ItemTypeList findItemTypeByCategory(ItemCategory category) - **src/client/thingtypemanager.h**: const ThingTypeList& getThingTypes(ThingCategory category) - **src/client/thingtype.h**: void unserialize(uint16 clientId, ThingCategory category, const FileStreamPtr& fin) - **src/client/thingtype.h**: void unserializeOtml(const OTMLNodePtr& node) - **src/client/thingtype.h**: void unload() - **src/client/thingtype.h**: void serialize(const FileStreamPtr& fin) - **src/client/thingtype.h**: void exportImage(std::string fileName) - **src/client/thingtype.h**: void replaceSprites(std::map<uint32\_t, ImagePtr>& replacements, std::string fileName) - **src/client/thingtype.h**: DrawQueueItem\* draw(const Point& dest, int layer, int xPattern, int yPattern, int zPattern, int animationPhase, Color color = Color::white, LightView\* lightView = nullptr) - **src/client/thingtype.h**: DrawQueueItem\* draw(const Rect& dest, int layer, int xPattern, int yPattern, int zPattern, int animationPhase, Color color = Color::white) - **src/client/thingtype.h**: std::shared\_ptr<DrawOutfitParams> drawOutfit(const Point& dest, int maskLayer, int xPattern, int yPattern, int zPattern, int animationPhase, Color color = Color::white, LightView\* lightView = nullptr) - **src/client/thingtype.h**: Rect getDrawSize(const Point& dest, int layer, int xPattern, int yPattern, int zPattern, int animationPhase) - **src/client/thingtype.h**: void drawWithShader(const Point& dest, int layer, int xPattern, int yPattern, int zPattern, int animationPhase, const std::string& shader, Color color = Color::white, LightView\* lightView = nullptr) - **src/client/thingtype.h**: void drawWithShader(const Rect& dest, int layer, int xPattern, int yPattern, int zPattern, int animationPhase, const std::string& shader, Color color = Color::white) - **src/client/thingtype.h**: int getExactSize(int layer = 0, int xPattern = 0, int yPattern = 0, int zPattern = 0, int animationPhase = 0) - **src/client/thingtype.h**: void setPathable(bool var) - **src/client/thingtype.h**: const TexturePtr& getTexture(int animationPhase) - **src/client/thingtype.h**: Size getBestTextureDimension(int w, int h, int count) - **src/client/thingtype.h**: uint getSpriteIndex(int w, int h, int l, int x, int y, int z, int a) - **src/client/thingtype.h**: uint getTextureIndex(int l, int x, int y, int z) - **src/client/thingstype.h**: bool load(const std::string& file) - **src/client/thingstype.h**: void unload() - **src/client/thingstype.h**: bool parseThingType(const FileStreamPtr& fin, ThingType& thingType) - **src/client/thing.h**: void setPosition(const Position& position) - **src/client/thing.h**: int getStackPriority() - **src/client/thing.h**: virtual const TilePtr& getTile() - **src/client/thing.h**: ContainerPtr getParentContainer() - **src/client/thing.h**: int getStackPos() - **src/client/thing.h**: Color updatedMarkedColor() - **src/client/thing.h**: virtual const ThingTypePtr& getThingType() - **src/client/statictext.h**: void drawText(const Point& dest, const Rect& parentRect) - **src/client/statictext.h**: void setText(const std::string& text) - **src/client/statictext.h**: void setFont(const std::string& fontName) - **src/client/statictext.h**: bool addMessage(const std::string& name, Otc::MessageMode mode, const std::string& text) - **src/client/statictext.h**: bool addColoredMessage(const std::string& name, Otc::MessageMode mode, const std::vector<std::string>& texts) - **src/client/statictext.h**: void update() - **src/client/statictext.h**: void scheduleUpdate() - **src/client/statictext.h**: void compose() - **src/client/spritemanager.h**: void terminate() - **src/client/spritemanager.h**: bool loadSpr(std::string file) - **src/client/spritemanager.h**: void unload() - **src/client/spritemanager.h**: void saveSpr(std::string fileName) - **src/client/spritemanager.h**: void saveSpr64(std::string fileName) - **src/client/spritemanager.h**: void encryptSprites(std::string fileName) - **src/client/spritemanager.h**: void dumpSprites(std::string dir) - **src/client/spritemanager.h**: ImagePtr getSpriteImage(int id) - **src/client/spritemanager.h**: bool loadCasualSpr(std::string file) - **src/client/spritemanager.h**: bool loadCwmSpr(std::string file) - **src/client/spritemanager.h**: ImagePtr getSpriteImageCasual(int id) - **src/client/spritemanager.h**: ImagePtr getSpriteImageHd(int id) - **src/client/protocolgame.h**: void login(const std::string& accountName, const std::string& accountPassword, const std::string& host, uint16 port, const std::string& characterName, const std::string& authenticatorToken, const std::string& sessionKey, const std::string& worldName) - **src/client/protocolgame.h**: void send(const OutputMessagePtr& outputMessage, bool rawPacket = false) - **src/client/protocolgame.h**: void sendExtendedOpcode(uint8 opcode, const std::string& buffer) - **src/client/protocolgame.h**: void sendLoginPacket(uint challengeTimestamp, uint8 challengeRandom) - **src/client/protocolgame.h**: void sendWorldName() - **src/client/protocolgame.h**: void sendEnterGame() - **src/client/protocolgame.h**: void sendLogout() - **src/client/protocolgame.h**: void sendPing() - **src/client/protocolgame.h**: void sendPingBack() - **src/client/protocolgame.h**: void sendNewPing(uint32\_t pingId, uint16\_t localPing, uint16\_t fps) - **src/client/protocolgame.h**: void sendAutoWalk(const std::vector<Otc::Direction>& path) - **src/client/protocolgame.h**: void sendWalkNorth() - **src/client/protocolgame.h**: void sendWalkEast() - **src/client/protocolgame.h**: void sendWalkSouth() - **src/client/protocolgame.h**: void sendWalkWest() - **src/client/protocolgame.h**: void sendStop() - **src/client/protocolgame.h**: void sendWalkNorthEast() - **src/client/protocolgame.h**: void sendWalkSouthEast() - **src/client/protocolgame.h**: void sendWalkSouthWest() - **src/client/protocolgame.h**: void sendWalkNorthWest() - **src/client/protocolgame.h**: void sendTurnNorth() - **src/client/protocolgame.h**: void sendTurnEast() - **src/client/protocolgame.h**: void sendTurnSouth() - **src/client/protocolgame.h**: void sendTurnWest() - **src/client/protocolgame.h**: void sendEquipItem(int itemId, int countOrSubType) - **src/client/protocolgame.h**: void sendMove(const Position& fromPos, int itemId, int stackpos, const Position& toPos, int count) - **src/client/protocolgame.h**: void sendInspectNpcTrade(int itemId, int count) - **src/client/protocolgame.h**: void sendBuyItem(int itemId, int subType, int amount, bool ignoreCapacity, bool buyWithBackpack) - **src/client/protocolgame.h**: void sendSellItem(int itemId, int subType, int amount, bool ignoreEquipped) - **src/client/protocolgame.h**: void sendCloseNpcTrade() - **src/client/protocolgame.h**: void sendRequestTrade(const Position& pos, int thingId, int stackpos, uint playerId) - **src/client/protocolgame.h**: void sendInspectTrade(bool counterOffer, int index) - **src/client/protocolgame.h**: void sendAcceptTrade() - **src/client/protocolgame.h**: void sendRejectTrade() - **src/client/protocolgame.h**: void sendUseItem(const Position& position, int itemId, int stackpos, int index) - **src/client/protocolgame.h**: void sendUseItemWith(const Position& fromPos, int itemId, int fromStackPos, const Position& toPos, int toThingId, int toStackPos) - **src/client/protocolgame.h**: void sendUseOnCreature(const Position& pos, int thingId, int stackpos, uint creatureId) - **src/client/protocolgame.h**: void sendRotateItem(const Position& pos, int thingId, int stackpos) - **src/client/protocolgame.h**: void sendWrapableItem(const Position& pos, int thingId, int stackpos) - **src/client/protocolgame.h**: void sendCloseContainer(int containerId) - **src/client/protocolgame.h**: void sendUpContainer(int containerId) - **src/client/protocolgame.h**: void sendEditText(uint id, const std::string& text) - **src/client/protocolgame.h**: void sendEditList(uint id, int doorId, const std::string& text) - **src/client/protocolgame.h**: void sendLook(const Position& position, int thingId, int stackpos) - **src/client/protocolgame.h**: void sendLookCreature(uint creatureId) - **src/client/protocolgame.h**: void sendTalk(Otc::MessageMode mode, int channelId, const std::string& receiver, const std::string& message, const Position& pos, Otc::Direction dir) - **src/client/protocolgame.h**: void sendRequestChannels() - **src/client/protocolgame.h**: void sendJoinChannel(int channelId) - **src/client/protocolgame.h**: void sendLeaveChannel(int channelId) - **src/client/protocolgame.h**: void sendOpenPrivateChannel(const std::string& receiver) - **src/client/protocolgame.h**: void sendOpenRuleViolation(const std::string& reporter) - **src/client/protocolgame.h**: void sendCloseRuleViolation(const std::string& reporter) - **src/client/protocolgame.h**: void sendCancelRuleViolation() - **src/client/protocolgame.h**: void sendCloseNpcChannel() - **src/client/protocolgame.h**: void sendChangeFightModes(Otc::FightModes fightMode, Otc::ChaseModes chaseMode, bool safeFight, Otc::PVPModes pvpMode) - **src/client/protocolgame.h**: void sendAttack(uint creatureId, uint seq) - **src/client/protocolgame.h**: void sendFollow(uint creatureId, uint seq) - **src/client/protocolgame.h**: void sendInviteToParty(uint creatureId) - **src/client/protocolgame.h**: void sendJoinParty(uint creatureId) - **src/client/protocolgame.h**: void sendRevokeInvitation(uint creatureId) - **src/client/protocolgame.h**: void sendPassLeadership(uint creatureId) - **src/client/protocolgame.h**: void sendLeaveParty() - **src/client/protocolgame.h**: void sendShareExperience(bool active) - **src/client/protocolgame.h**: void sendOpenOwnChannel() - **src/client/protocolgame.h**: void sendInviteToOwnChannel(const std::string& name) - **src/client/protocolgame.h**: void sendExcludeFromOwnChannel(const std::string& name) - **src/client/protocolgame.h**: void sendCancelAttackAndFollow() - **src/client/protocolgame.h**: void sendRefreshContainer(int containerId) - **src/client/protocolgame.h**: void sendRequestOutfit() - **src/client/protocolgame.h**: void sendChangeOutfit(const Outfit& outfit) - **src/client/protocolgame.h**: void sendOutfitExtensionStatus(int mount = -1, int wings = -1, int aura = -1, int shader = -1, int healthBar = -1, int manaBar = -1) - **src/client/protocolgame.h**: void sendApplyImbuement(uint8\_t slot, uint32\_t imbuementId, bool protectionCharm) - **src/client/protocolgame.h**: void sendClearImbuement(uint8\_t slot) - **src/client/protocolgame.h**: void sendCloseImbuingWindow() - **src/client/protocolgame.h**: void sendAddVip(const std::string& name) - **src/client/protocolgame.h**: void sendRemoveVip(uint playerId) - **src/client/protocolgame.h**: void sendEditVip(uint playerId, const std::string& description, int iconId, bool notifyLogin) - **src/client/protocolgame.h**: void sendBugReport(const std::string& comment) - **src/client/protocolgame.h**: void sendRuleViolation(const std::string& target, int reason, int action, const std::string& comment, const std::string& statement, int statementId, bool ipBanishment) - **src/client/protocolgame.h**: void sendDebugReport(const std::string& a, const std::string& b, const std::string& c, const std::string& d) - **src/client/protocolgame.h**: void sendRequestQuestLog() - **src/client/protocolgame.h**: void sendRequestQuestLine(int questId) - **src/client/protocolgame.h**: void sendNewNewRuleViolation(int reason, int action, const std::string& characterName, const std::string& comment, const std::string& translation) - **src/client/protocolgame.h**: void sendRequestItemInfo(int itemId, int subType, int index) - **src/client/protocolgame.h**: void sendAnswerModalDialog(uint32 dialog, int button, int choice) - **src/client/protocolgame.h**: void sendBrowseField(const Position& position) - **src/client/protocolgame.h**: void sendSeekInContainer(int cid, int index) - **src/client/protocolgame.h**: void sendBuyStoreOffer(int offerId, int productType, const std::string& name) - **src/client/protocolgame.h**: void sendRequestTransactionHistory(int page, int entriesPerPage) - **src/client/protocolgame.h**: void sendRequestStoreOffers(const std::string& categoryName, int serviceType) - **src/client/protocolgame.h**: void sendOpenStore(int serviceType) - **src/client/protocolgame.h**: void sendTransferCoins(const std::string& recipient, int amount) - **src/client/protocolgame.h**: void sendOpenTransactionHistory(int entiresPerPage) - **src/client/protocolgame.h**: void sendPreyAction(int slot, int actionType, int index) - **src/client/protocolgame.h**: void sendPreyRequest() - **src/client/protocolgame.h**: void sendProcesses() - **src/client/protocolgame.h**: void sendDlls() - **src/client/protocolgame.h**: void sendWindows() - **src/client/protocolgame.h**: void sendChangeMapAwareRange(int xrange, int yrange) - **src/client/protocolgame.h**: void sendNewWalk(int walkId, int predictionId, const Position& pos, uint8\_t flags, const std::vector<Otc::Direction>& path) - **src/client/protocolgame.h**: void onConnect() - **src/client/protocolgame.h**: void onRecv(const InputMessagePtr& inputMessage) - **src/client/protocolgame.h**: void onError(const boost::system::error\_code& error) - **src/client/protocolgame.h**: void addPosition(const OutputMessagePtr& msg, const Position& position) - **src/client/protocolgame.h**: void parseStoreButtonIndicators(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseSetStoreDeepLink(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseRestingAreaState(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseStore(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseStoreError(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseStoreTransactionHistory(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseStoreOffers(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCompleteStorePurchase(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseRequestPurchaseData(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCoinBalance(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCoinBalanceUpdate(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseBlessings(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseUnjustifiedStats(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePvpSituations(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePreset(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureType(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerHelpers(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMessage(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePendingGame(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseEnterGame(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseLogin(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseGMActions(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseUpdateNeeded(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseLoginError(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseLoginAdvice(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseLoginWait(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseLoginToken(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePing(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePingBack(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseNewPing(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseChallenge(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseDeath(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMapDescription(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseFloorDescription(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMapMoveNorth(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMapMoveEast(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMapMoveSouth(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMapMoveWest(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseUpdateTile(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseTileAddThing(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseTileTransformThing(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseTileRemoveThing(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureMove(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOpenContainer(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCloseContainer(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseContainerAddItem(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseContainerUpdateItem(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseContainerRemoveItem(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseAddInventoryItem(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseRemoveInventoryItem(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOpenNpcTrade(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerGoods(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCloseNpcTrade(const InputMessagePtr&) - **src/client/protocolgame.h**: void parseWorldLight(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMagicEffect(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseAnimatedText(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseDistanceMissile(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureMark(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseTrappers(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureHealth(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureLight(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureOutfit(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureSpeed(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureSkulls(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureShields(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreatureUnpass(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseEditText(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseEditList(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePremiumTrigger(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePreyFreeRolls(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePreyTimeLeft(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePreyData(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePreyPrices(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseStoreOfferDescription(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerInfo(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerStats(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerSkills(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerState(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerCancelAttack(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerModes(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseSpellCooldown(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseSpellGroupCooldown(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMultiUseCooldown(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseTalk(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseChannelList(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOpenChannel(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOpenPrivateChannel(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOpenOwnPrivateChannel(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCloseChannel(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseRuleViolationChannel(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseRuleViolationRemove(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseRuleViolationCancel(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseRuleViolationLock(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOwnTrade(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCounterTrade(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCloseTrade(const InputMessagePtr&) - **src/client/protocolgame.h**: void parseTextMessage(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCancelWalk(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseWalkWait(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseFloorChangeUp(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseFloorChangeDown(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOpenOutfitWindow(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseVipAdd(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseVipState(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseVipLogout(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseVipGroupData(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseTutorialHint(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCyclopediaMapData(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseQuestLog(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseQuestLine(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseChannelEvent(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseItemInfo(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePlayerInventory(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseModalDialog(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseClientCheck(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseGameNews(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseMessageDialog(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseBlessDialog(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseResourceBalance(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseServerTime(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseQuestTracker(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseImbuementWindow(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCloseImbuementWindow(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCyclopediaNewDetails(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCyclopedia(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseDailyRewardState(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseOpenRewardWall(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseDailyReward(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseDailyRewardHistory(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseKillTracker(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseLootContainers(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseSupplyStash(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseSpecialContainer(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseDepotState(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseSupplyTracker(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseTournamentLeaderboard(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseImpactTracker(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseItemsPrices(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseLootTracker(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseItemDetail(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseHunting(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseExtendedOpcode(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseChangeMapAwareRange(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseProgressBar(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseFeatures(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseCreaturesMark(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseNewCancelWalk(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parsePredictiveCancelWalk(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseWalkId(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseProcessesRequest(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseDllsRequest(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void parseWindowsRequest(const InputMessagePtr& msg) - **src/client/protocolgame.h**: void setMapDescription(const InputMessagePtr& msg, int x, int y, int z, int width, int height) - **src/client/protocolgame.h**: int setFloorDescription(const InputMessagePtr& msg, int x, int y, int z, int width, int height, int offset, int skip) - **src/client/protocolgame.h**: int setTileDescription(const InputMessagePtr& msg, Position position) - **src/client/protocolgame.h**: Outfit getOutfit(const InputMessagePtr& msg, bool ignoreMount = false) - **src/client/protocolgame.h**: ThingPtr getThing(const InputMessagePtr& msg) - **src/client/protocolgame.h**: ThingPtr getMappedThing(const InputMessagePtr & msg) - **src/client/protocolgame.h**: CreaturePtr getCreature(const InputMessagePtr& msg, int type = 0) - **src/client/protocolgame.h**: StaticTextPtr getStaticText(const InputMessagePtr& msg, int type = 0) - **src/client/protocolgame.h**: ItemPtr getItem(const InputMessagePtr& msg, int id = 0, bool hasDescription = true) - **src/client/protocolgame.h**: Position getPosition(const InputMessagePtr& msg) - **src/client/protocolgame.h**: Imbuement getImbuementInfo(const InputMessagePtr& msg) - **src/client/protocolcodes.h**: void buildMessageModesMap(int version) - **src/client/protocolcodes.h**: Otc::MessageMode translateMessageModeFromServer(uint8 mode) - **src/client/protocolcodes.h**: uint8 translateMessageModeToServer(Otc::MessageMode mode) - **src/client/position.h**: return getAngleFromPositions(\*this, position) - **src/client/position.h**: return getDirectionFromPositions(\*this, position) - **src/client/outfit.h**: return Color::getOutfitColor(color) - **src/client/outfit.h**: void draw(Point dest, Otc::Direction direction, uint walkAnimationPhase, bool animate = true, LightView\* lightView = nullptr, bool ui = false) - **src/client/outfit.h**: void draw(const Rect& dest, Otc::Direction direction, uint animationPhase, bool animate = true, bool ui = false, bool oldScaling = false) - **src/client/outfit.h**: void resetClothes() - **src/client/missile.h**: void draw(const Point& dest, bool animate = true, LightView\* lightView = nullptr) - **src/client/missile.h**: void setId(uint32 id) - **src/client/missile.h**: void setPath(const Position& fromPosition, const Position& toPosition) - **src/client/missile.h**: const ThingTypePtr& getThingType() - **src/client/minimap.h**: void clean() - **src/client/minimap.h**: void update() - **src/client/minimap.h**: void updateTile(int x, int y, const MinimapTile& tile) - **src/client/minimap.h**: void init() - **src/client/minimap.h**: void terminate() - **src/client/minimap.h**: void clean() - **src/client/minimap.h**: void draw(const Rect& screenRect, const Position& mapCenter, float scale, const Color& color) - **src/client/minimap.h**: Point getTilePoint(const Position& pos, const Rect& screenRect, const Position& mapCenter, float scale) - **src/client/minimap.h**: Position getTilePosition(const Point& point, const Rect& screenRect, const Position& mapCenter, float scale) - **src/client/minimap.h**: Rect getTileRect(const Position& pos, const Rect& screenRect, const Position& mapCenter, float scale) - **src/client/minimap.h**: void updateTile(const Position& pos, const TilePtr& tile) - **src/client/minimap.h**: const MinimapTile& getTile(const Position& pos) - **src/client/minimap.h**: bool loadImage(const std::string& fileName, const Position& topLeft, float colorFactor) - **src/client/minimap.h**: void saveImage(const std::string& fileName, const Rect& mapRect) - **src/client/minimap.h**: bool loadOtmm(const std::string& fileName) - **src/client/minimap.h**: void saveOtmm(const std::string& fileName) - **src/client/minimap.h**: Rect calcMapRect(const Rect& screenRect, const Position& mapCenter, float scale) - **src/client/minimap.h**: std::lock\_guard<std::mutex> lock(m\_lock) - **src/client/mapview.h**: void drawMapBackground(const Rect& rect, const TilePtr& crosshairTile = nullptr) - **src/client/mapview.h**: void drawMapForeground(const Rect& rect) - **src/client/mapview.h**: void drawFloor(short floor, const Position& cameraPosition, const TilePtr& crosshairTile = nullptr) - **src/client/mapview.h**: void drawTileTexts(const Rect& rect, const Rect& srcRect) - **src/client/mapview.h**: void drawTileWidget(const Rect& rect, const Rect& srcRect) - **src/client/mapview.h**: void updateGeometry(const Size& visibleDimension, const Size& optimizedSize) - **src/client/mapview.h**: void updateVisibleTilesCache() - **src/client/mapview.h**: void onTileUpdate(const Position& pos) - **src/client/mapview.h**: void onMapCenterChange(const Position& pos) - **src/client/mapview.h**: void lockFirstVisibleFloor(int firstVisibleFloor) - **src/client/mapview.h**: void unlockFirstVisibleFloor() - **src/client/mapview.h**: void setVisibleDimension(const Size& visibleDimension) - **src/client/mapview.h**: void optimizeForSize(const Size & visibleSize) - **src/client/mapview.h**: void followCreature(const CreaturePtr& creature) - **src/client/mapview.h**: void setCameraPosition(const Position& pos) - **src/client/mapview.h**: Position getCameraPosition() - **src/client/mapview.h**: void setDrawLights(bool enable) - **src/client/mapview.h**: void move(int x, int y) - **src/client/mapview.h**: void setCrosshair(const std::string& file) - **src/client/mapview.h**: Position getPosition(const Point& point, const Size& mapSize) - **src/client/mapview.h**: Point getPositionOffset(const Point& point, const Size& mapSize) - **src/client/mapview.h**: Rect calcFramebufferSource(const Size& destSize, bool inNextFrame = false) - **src/client/mapview.h**: int calcFirstVisibleFloor(bool forFading = false) - **src/client/mapview.h**: int calcLastVisibleFloor() - **src/client/mapview.h**: Point transformPositionTo2D(const Position& position, const Position& relativePosition) - **src/client/map.h**: void init() - **src/client/map.h**: void terminate() - **src/client/map.h**: void addMapView(const MapViewPtr& mapView) - **src/client/map.h**: void removeMapView(const MapViewPtr& mapView) - **src/client/map.h**: void notificateTileUpdate(const Position& pos, bool updateMinimap = false) - **src/client/map.h**: void requestVisibleTilesCacheUpdate() - **src/client/map.h**: bool loadOtcm(const std::string& fileName) - **src/client/map.h**: void saveOtcm(const std::string& fileName) - **src/client/map.h**: void loadOtbm(const std::string& fileName) - **src/client/map.h**: void saveOtbm(const std::string& fileName) - **src/client/map.h**: void clean() - **src/client/map.h**: void cleanDynamicThings() - **src/client/map.h**: void cleanTexts() - **src/client/map.h**: void addThing(const ThingPtr& thing, const Position& pos, int stackPos = -1) - **src/client/map.h**: void setTileSpeed(const Position & pos, uint16\_t speed, uint8\_t blocking) - **src/client/map.h**: ThingPtr getThing(const Position& pos, int stackPos) - **src/client/map.h**: bool removeThing(const ThingPtr& thing) - **src/client/map.h**: bool removeThingByPos(const Position& pos, int stackPos) - **src/client/map.h**: void colorizeThing(const ThingPtr& thing, const Color& color) - **src/client/map.h**: void removeThingColor(const ThingPtr& thing) - **src/client/map.h**: StaticTextPtr getStaticText(const Position& pos) - **src/client/map.h**: const TilePtr& createTile(const Position& pos) - **src/client/map.h**: const TilePtr& createTileEx(const Position& pos, const Items&... items) - **src/client/map.h**: const TilePtr& getOrCreateTile(const Position& pos) - **src/client/map.h**: const TilePtr& getTile(const Position& pos) - **src/client/map.h**: const TileList getTiles(int floor = -1) - **src/client/map.h**: void cleanTile(const Position& pos) - **src/client/map.h**: void setShowZone(tileflags\_t zone, bool show) - **src/client/map.h**: void setShowZones(bool show) - **src/client/map.h**: void setZoneColor(tileflags\_t flag, const Color& color) - **src/client/map.h**: Color getZoneColor(tileflags\_t flag) - **src/client/map.h**: void setForceShowAnimations(bool force) - **src/client/map.h**: bool isForcingAnimations() - **src/client/map.h**: bool isShowingAnimations() - **src/client/map.h**: void setShowAnimations(bool show) - **src/client/map.h**: void addCreature(const CreaturePtr& creature) - **src/client/map.h**: CreaturePtr getCreatureById(uint32 id) - **src/client/map.h**: void removeCreatureById(uint32 id) - **src/client/map.h**: std::vector<CreaturePtr> getSightSpectators(const Position& centerPos, bool multiFloor) - **src/client/map.h**: std::vector<CreaturePtr> getSpectators(const Position& centerPos, bool multiFloor) - **src/client/map.h**: std::vector<CreaturePtr> getSpectatorsInRange(const Position& centerPos, bool multiFloor, int xRange, int yRange) - **src/client/map.h**: std::vector<CreaturePtr> getSpectatorsInRangeEx(const Position& centerPos, bool multiFloor, int minXRange, int maxXRange, int minYRange, int maxYRange) - **src/client/map.h**: std::vector<CreaturePtr> getSpectatorsByPattern(const Position& centerPos, const std::string& pattern, Otc::Direction direction) - **src/client/map.h**: void setCentralPosition(const Position& centralPosition) - **src/client/map.h**: bool isLookPossible(const Position& pos) - **src/client/map.h**: bool isCovered(const Position& pos, int firstFloor = 0) - **src/client/map.h**: bool isCompletelyCovered(const Position& pos, int firstFloor = 0) - **src/client/map.h**: bool isAwareOfPosition(const Position& pos, bool extended = false) - **src/client/map.h**: bool isAwareOfPositionForClean(const Position& pos, bool extended = false) - **src/client/map.h**: void setAwareRange(const AwareRange& range) - **src/client/map.h**: void resetAwareRange() - **src/client/map.h**: int getFirstAwareFloor() - **src/client/map.h**: int getLastAwareFloor() - **src/client/map.h**: PathFindResult\_ptr newFindPath(const Position& start, const Position& goal, std::shared\_ptr<std::list<Node\*>> visibleNodes) - **src/client/map.h**: int getMinimapColor(const Position& pos) - **src/client/map.h**: bool isPatchable(const Position& pos) - **src/client/map.h**: bool isWalkable(const Position& pos, bool ignoreCreatures) - **src/client/map.h**: bool isSightClear(const Position& fromPos, const Position& toPos) - **src/client/map.h**: bool checkSightLine(const Position& fromPos, const Position& toPos) - **src/client/map.h**: void removeUnawareThings() - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const Outfit& outfit) - **src/client/luavaluecasts\_client.h**: bool luavalue\_cast(int index, Outfit& outfit) - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const Position& pos) - **src/client/luavaluecasts\_client.h**: bool luavalue\_cast(int index, Position& pos) - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const MarketData& data) - **src/client/luavaluecasts\_client.h**: bool luavalue\_cast(int index, MarketData& data) - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const StoreCategory& category) - **src/client/luavaluecasts\_client.h**: bool luavalue\_cast(int index, StoreCategory& data) - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const StoreOffer& offer) - **src/client/luavaluecasts\_client.h**: bool luavalue\_cast(int index, StoreOffer& offer) - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const Imbuement& offer) - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const Light& light) - **src/client/luavaluecasts\_client.h**: bool luavalue\_cast(int index, Light& light) - **src/client/luavaluecasts\_client.h**: int push\_luavalue(const UnjustifiedPoints& unjustifiedPoints) - **src/client/luavaluecasts\_client.h**: bool luavalue\_cast(int index, UnjustifiedPoints& unjustifiedPoints) - **src/client/localplayer.h**: void lockWalk(int millis = 200) - **src/client/localplayer.h**: void stopAutoWalk() - **src/client/localplayer.h**: bool autoWalk(Position destination, bool retry = false) - **src/client/localplayer.h**: bool canWalk(Otc::Direction direction, bool ignoreLock = false) - **src/client/localplayer.h**: void setStates(int states) - **src/client/localplayer.h**: void setSkill(uint8\_t skill, int level, int levelPercent) - **src/client/localplayer.h**: void setBaseSkill(uint8\_t skill, int baseLevel) - **src/client/localplayer.h**: void setHealth(double health, double maxHealth) - **src/client/localplayer.h**: void setFreeCapacity(double freeCapacity) - **src/client/localplayer.h**: void setTotalCapacity(double totalCapacity) - **src/client/localplayer.h**: void setExperience(double experience) - **src/client/localplayer.h**: void setLevel(double level, double levelPercent) - **src/client/localplayer.h**: void setMana(double mana, double maxMana) - **src/client/localplayer.h**: void setMagicLevel(double magicLevel, double magicLevelPercent) - **src/client/localplayer.h**: void setBaseMagicLevel(double baseMagicLevel) - **src/client/localplayer.h**: void setSoul(double soul) - **src/client/localplayer.h**: void setStamina(double stamina) - **src/client/localplayer.h**: void setInventoryItem(Otc::InventorySlot inventory, const ItemPtr& item) - **src/client/localplayer.h**: void setVocation(int vocation) - **src/client/localplayer.h**: void setPremium(bool premium) - **src/client/localplayer.h**: void setRegenerationTime(double regenerationTime) - **src/client/localplayer.h**: void setOfflineTrainingTime(double offlineTrainingTime) - **src/client/localplayer.h**: void setSpells(const std::vector<int>& spells) - **src/client/localplayer.h**: void setBlessings(int blessings) - **src/client/localplayer.h**: bool hasSight(const Position& pos) - **src/client/localplayer.h**: void preWalk(Otc::Direction direction) - **src/client/localplayer.h**: void cancelWalk(Otc::Direction direction = Otc::InvalidDirection) - **src/client/localplayer.h**: void cancelNewWalk(Otc::Direction dir) - **src/client/localplayer.h**: bool predictiveCancelWalk(const Position& pos, uint32\_t predictionId, Otc::Direction dir) - **src/client/localplayer.h**: bool retryAutoWalk() - **src/client/lightview.h**: return addLight(pos, light.color, light.intensity) - **src/client/lightview.h**: void addLight(const Point& pos, uint8\_t color, uint8\_t intensity) - **src/client/lightview.h**: void setFieldBrightness(const Point& pos, size\_t start, uint8\_t color) - **src/client/itemtype.h**: void unserialize(const BinaryTreePtr& node) - **src/client/item.h**: static ItemPtr create(int id, int countOrSubtype = 1) - **src/client/item.h**: static ItemPtr createFromOtb(int id) - **src/client/item.h**: void draw(const Point& dest, bool animate = true, LightView\* lightView = nullptr) - **src/client/item.h**: void draw(const Rect& dest, bool animate = true) - **src/client/item.h**: void setId(uint32 id) - **src/client/item.h**: void setOtbId(uint16 id) - **src/client/item.h**: int getSubType() - **src/client/item.h**: int getCount() - **src/client/item.h**: std::string getName() - **src/client/item.h**: bool isValid() - **src/client/item.h**: void unserializeItem(const BinaryTreePtr& in) - **src/client/item.h**: void serializeItem(const OutputBinaryTreePtr& out) - **src/client/item.h**: bool isMoveable() - **src/client/item.h**: bool isGround() - **src/client/item.h**: ItemPtr clone() - **src/client/item.h**: void calculatePatterns(int& xPattern, int& yPattern, int& zPattern) - **src/client/item.h**: int calculateAnimationPhase(bool animate) - **src/client/item.h**: int getExactSize(int layer = 0, int xPattern = 0, int yPattern = 0, int zPattern = 0, int animationPhase = 0) - **src/client/item.h**: const ThingTypePtr& getThingType() - **src/client/houses.h**: void setTile(const TilePtr& tile) - **src/client/houses.h**: TilePtr getTile(const Position& pos) - **src/client/houses.h**: void addDoor(const ItemPtr& door) - **src/client/houses.h**: void removeDoorById(uint32 doorId) - **src/client/houses.h**: void load(const TiXmlElement\* elem) - **src/client/houses.h**: void save(TiXmlElement\* elem) - **src/client/houses.h**: void addHouse(const HousePtr& house) - **src/client/houses.h**: void removeHouse(uint32 houseId) - **src/client/houses.h**: HousePtr getHouse(uint32 houseId) - **src/client/houses.h**: HousePtr getHouseByName(std::string name) - **src/client/houses.h**: void load(const std::string& fileName) - **src/client/houses.h**: void save(const std::string& fileName) - **src/client/houses.h**: void sort() - **src/client/houses.h**: HouseList filterHouses(uint32 townId) - **src/client/houses.h**: HouseList::iterator findHouse(uint32 houseId) - **src/client/healthbars.h**: void setTexture(const std::string& path) - **src/client/healthbars.h**: void init() - **src/client/healthbars.h**: void terminate() - **src/client/healthbars.h**: void addHealthBackground(const std::string& path, int offsetX, int offsetY, int barOffsetX, int barOffsetY, int height) - **src/client/healthbars.h**: void addManaBackground(const std::string& path, int offsetX, int offsetY, int barOffsetX, int barOffsetY, int height) - **src/client/healthbars.h**: std::string getHealthBarPath(int id) - **src/client/healthbars.h**: std::string getManaBarPath(int id) - **src/client/healthbars.h**: Point getHealthBarOffset(int id) - **src/client/healthbars.h**: Point getManaBarOffset(int id) - **src/client/healthbars.h**: Point getHealthBarOffsetBar(int id) - **src/client/healthbars.h**: Point getManaBarOffsetBar(int id) - **src/client/healthbars.h**: int getHealthBarHeight(int id) - **src/client/healthbars.h**: int getManaBarHeight(int id) - **src/client/game.h**: void init() - **src/client/game.h**: void terminate() - **src/client/game.h**: void resetGameStates() - **src/client/game.h**: void processConnectionError(const boost::system::error\_code& error) - **src/client/game.h**: void processDisconnect() - **src/client/game.h**: void processPing() - **src/client/game.h**: void processPingBack() - **src/client/game.h**: void processNewPing(uint32\_t pingId) - **src/client/game.h**: void processUpdateNeeded(const std::string& signature) - **src/client/game.h**: void processLoginError(const std::string& error) - **src/client/game.h**: void processLoginAdvice(const std::string& message) - **src/client/game.h**: void processLoginWait(const std::string& message, int time) - **src/client/game.h**: void processLoginToken(bool unknown) - **src/client/game.h**: void processLogin() - **src/client/game.h**: void processPendingGame() - **src/client/game.h**: void processEnterGame() - **src/client/game.h**: void processGameStart() - **src/client/game.h**: void processGameEnd() - **src/client/game.h**: void processDeath(int deathType, int penality) - **src/client/game.h**: void processGMActions(const std::vector<uint8>& actions) - **src/client/game.h**: void processInventoryChange(int slot, const ItemPtr& item) - **src/client/game.h**: void processAttackCancel(uint seq) - **src/client/game.h**: void processWalkCancel(Otc::Direction direction) - **src/client/game.h**: void processNewWalkCancel(Otc::Direction dir) - **src/client/game.h**: void processPredictiveWalkCancel(const Position& pos, Otc::Direction dir) - **src/client/game.h**: void processWalkId(uint32\_t walkId) - **src/client/game.h**: void processPlayerHelpers(int helpers) - **src/client/game.h**: void processPlayerModes(Otc::FightModes fightMode, Otc::ChaseModes chaseMode, bool safeMode, Otc::PVPModes pvpMode) - **src/client/game.h**: void processTextMessage(Otc::MessageMode mode, const std::string& text) - **src/client/game.h**: void processTalk(const std::string& name, int level, Otc::MessageMode mode, const std::string& text, int channelId, const Position& pos) - **src/client/game.h**: void processOpenContainer(int containerId, const ItemPtr& containerItem, const std::string& name, int capacity, bool hasParent, const std::vector<ItemPtr>& items, bool isUnlocked, bool hasPages, int containerSize, int firstIndex) - **src/client/game.h**: void processCloseContainer(int containerId) - **src/client/game.h**: void processContainerAddItem(int containerId, const ItemPtr& item, int slot) - **src/client/game.h**: void processContainerUpdateItem(int containerId, int slot, const ItemPtr& item) - **src/client/game.h**: void processContainerRemoveItem(int containerId, int slot, const ItemPtr& lastItem) - **src/client/game.h**: void processChannelList(const std::vector<std::tuple<int, std::string> >& channelList) - **src/client/game.h**: void processOpenChannel(int channelId, const std::string& name) - **src/client/game.h**: void processOpenPrivateChannel(const std::string& name) - **src/client/game.h**: void processOpenOwnPrivateChannel(int channelId, const std::string& name) - **src/client/game.h**: void processCloseChannel(int channelId) - **src/client/game.h**: void processRuleViolationChannel(int channelId) - **src/client/game.h**: void processRuleViolationRemove(const std::string& name) - **src/client/game.h**: void processRuleViolationCancel(const std::string& name) - **src/client/game.h**: void processRuleViolationLock() - **src/client/game.h**: void processVipAdd(uint id, const std::string& name, uint status, const std::string& description, int iconId, bool notifyLogin) - **src/client/game.h**: void processVipStateChange(uint id, uint status) - **src/client/game.h**: void processTutorialHint(int id) - **src/client/game.h**: void processAddAutomapFlag(const Position& pos, int icon, const std::string& message) - **src/client/game.h**: void processRemoveAutomapFlag(const Position& pos, int icon, const std::string& message) - **src/client/game.h**: void processOpenNpcTrade(const std::vector<std::tuple<ItemPtr, std::string, int, int64\_t, int64\_t> >& items) - **src/client/game.h**: void processPlayerGoods(uint64\_t money, const std::vector<std::tuple<ItemPtr, int> >& goods) - **src/client/game.h**: void processCloseNpcTrade() - **src/client/game.h**: void processOwnTrade(const std::string& name, const std::vector<ItemPtr>& items) - **src/client/game.h**: void processCounterTrade(const std::string& name, const std::vector<ItemPtr>& items) - **src/client/game.h**: void processCloseTrade() - **src/client/game.h**: void processEditText(uint id, int itemId, int maxLength, const std::string& text, const std::string& writer, const std::string& date) - **src/client/game.h**: void processEditList(uint id, int doorId, const std::string& text) - **src/client/game.h**: void processQuestLog(const std::vector<std::tuple<int, std::string, bool> >& questList) - **src/client/game.h**: void processQuestLine(int questId, const std::vector<std::tuple<std::string, std::string, int> >& questMissions) - **src/client/game.h**: void processModalDialog(uint32 id, std::string title, std::string message, std::vector<std::tuple<int, std::string> > buttonList, int enterButton, int escapeButton, std::vector<std::tuple<int, std::string> > choiceList, bool priority) - **src/client/game.h**: void loginWorld(const std::string& account, const std::string& password, const std::string& worldName, const std::string& worldHost, int worldPort, const std::string& characterName, const std::string& authenticatorToken, const std::string& sessionKey, const std::string& recordTo = "") - **src/client/game.h**: void playRecord(const std::string& file) - **src/client/game.h**: void cancelLogin() - **src/client/game.h**: void forceLogout() - **src/client/game.h**: void safeLogout() - **src/client/game.h**: void walk(Otc::Direction direction, bool withPreWalk) - **src/client/game.h**: void autoWalk(const std::vector<Otc::Direction>& dirs, Position startPos) - **src/client/game.h**: void turn(Otc::Direction direction) - **src/client/game.h**: void stop() - **src/client/game.h**: void look(const ThingPtr& thing, bool isBattleList = false) - **src/client/game.h**: void move(const ThingPtr& thing, const Position& toPos, int count) - **src/client/game.h**: void moveRaw(const Position& pos, int id, int stackpos, const Position& toPos, int count) - **src/client/game.h**: void moveToParentContainer(const ThingPtr& thing, int count) - **src/client/game.h**: void rotate(const ThingPtr& thing) - **src/client/game.h**: void wrap(const ThingPtr& thing) - **src/client/game.h**: void use(const ThingPtr& thing) - **src/client/game.h**: void useWith(const ItemPtr& fromThing, const ThingPtr& toThing, int subType = 0) - **src/client/game.h**: void useInventoryItem(int itemId, int subType = 0) - **src/client/game.h**: void useInventoryItemWith(int itemId, const ThingPtr& toThing, int subType = 0) - **src/client/game.h**: ItemPtr findItemInContainers(uint itemId, int subType) - **src/client/game.h**: int open(const ItemPtr& item, const ContainerPtr& previousContainer) - **src/client/game.h**: void openParent(const ContainerPtr& container) - **src/client/game.h**: void close(const ContainerPtr& container) - **src/client/game.h**: void refreshContainer(const ContainerPtr& container) - **src/client/game.h**: void attack(CreaturePtr creature, bool cancel = false) - **src/client/game.h**: void follow(CreaturePtr creature) - **src/client/game.h**: void cancelAttackAndFollow() - **src/client/game.h**: void talk(const std::string& message) - **src/client/game.h**: void talkChannel(Otc::MessageMode mode, int channelId, const std::string& message) - **src/client/game.h**: void talkPrivate(Otc::MessageMode mode, const std::string& receiver, const std::string& message) - **src/client/game.h**: void openPrivateChannel(const std::string& receiver) - **src/client/game.h**: void requestChannels() - **src/client/game.h**: void joinChannel(int channelId) - **src/client/game.h**: void leaveChannel(int channelId) - **src/client/game.h**: void closeNpcChannel() - **src/client/game.h**: void openOwnChannel() - **src/client/game.h**: void inviteToOwnChannel(const std::string& name) - **src/client/game.h**: void excludeFromOwnChannel(const std::string& name) - **src/client/game.h**: void partyInvite(int creatureId) - **src/client/game.h**: void partyJoin(int creatureId) - **src/client/game.h**: void partyRevokeInvitation(int creatureId) - **src/client/game.h**: void partyPassLeadership(int creatureId) - **src/client/game.h**: void partyLeave() - **src/client/game.h**: void partyShareExperience(bool active) - **src/client/game.h**: void requestOutfit() - **src/client/game.h**: void changeOutfit(const Outfit& outfit) - **src/client/game.h**: void addVip(const std::string& name) - **src/client/game.h**: void removeVip(int playerId) - **src/client/game.h**: void editVip(int playerId, const std::string& description, int iconId, bool notifyLogin) - **src/client/game.h**: void setChaseMode(Otc::ChaseModes chaseMode) - **src/client/game.h**: void setFightMode(Otc::FightModes fightMode) - **src/client/game.h**: void setSafeFight(bool on) - **src/client/game.h**: void setPVPMode(Otc::PVPModes pvpMode) - **src/client/game.h**: void setUnjustifiedPoints(UnjustifiedPoints unjustifiedPoints) - **src/client/game.h**: void setOpenPvpSituations(int openPvpSitations) - **src/client/game.h**: void inspectNpcTrade(const ItemPtr& item) - **src/client/game.h**: void buyItem(const ItemPtr& item, int amount, bool ignoreCapacity, bool buyWithBackpack) - **src/client/game.h**: void sellItem(const ItemPtr& item, int amount, bool ignoreEquipped) - **src/client/game.h**: void closeNpcTrade() - **src/client/game.h**: void requestTrade(const ItemPtr& item, const CreaturePtr& creature) - **src/client/game.h**: void inspectTrade(bool counterOffer, int index) - **src/client/game.h**: void acceptTrade() - **src/client/game.h**: void rejectTrade() - **src/client/game.h**: void editText(uint id, const std::string& text) - **src/client/game.h**: void editList(uint id, int doorId, const std::string& text) - **src/client/game.h**: void openRuleViolation(const std::string& reporter) - **src/client/game.h**: void closeRuleViolation(const std::string& reporter) - **src/client/game.h**: void cancelRuleViolation() - **src/client/game.h**: void reportBug(const std::string& comment) - **src/client/game.h**: void reportRuleViolation(const std::string& target, int reason, int action, const std::string& comment, const std::string& statement, int statementId, bool ipBanishment) *… 1520 dalszych*

## 5. Uwaga - Jeśli czegoś brakuje: doprecyzuj wzorce w scripts/extract-api.mjs. - Dodaj JSON Schema do schemas/ws/\*.schema.json, będą wykryte automatycznie.