SA-MP Includes Library+ 1.1 Documentation

Ivan Kmeťo

August 2020

Contents

1 Changes in ISAMPP 1.1

ISAMPP Library:
- New function *ISAMPP_SHOWPLAYERPOSITION*
- Functions *ISAMPP_TELEPORT* and *ISAMPP_TELEPORTEX* are now part of library itself
- Fixed some Location Identifier coordinates
- Updated documentation

ISAMPP Sandbox Game Mode:
- Changed in-game teleport commands to use *ISAMPP_TELEPORTEX* function
- Changed in-game command *"/teleport [ID]"* to *"/tp [ID]"*
- In-game command *"/stringcols"* takes no additional parameters anymore
- New in-game command *"/showplayerpos"* works with *ISAMPP_SHOWPLAYERPOSITION* data

2   Introduction

San Andreas Multiplayer Includes Library+ or ISAMPP *(Includes for San Andreas Multiplayer Plus)* is library of include files for San Andreas Multiplayer Mod SA-MP. Purpose of ISAMPP is to make development of game modes for SA-MP much easier. SA-MP by default uses in most cases numeric identifiers which are hard to remember and without enough experience you have to use SA-MP wiki a lot. I believe that word identifiers are much more easier to remember, they make much more sense (especially if you are familiar with modding the game Grand Theft Auto: San Andreas) and may also have positive impact on your workflow. Except re-definitions of numeric identifiers, ISAMPP also contains other useful libraries - such as library of locations with coordinates, interior identifiers and location names - or few related custom stock functions. ISAMPP is available with sandbox-styled game mode in which you can see everything implemented. You are free to modify and distribute the source code accordingly to its license.

Note: ISAMPP is not part of San Andreas Multiplayer mod (SA-MP) and it is not affiliated with Rockstar Games, Rockstar North or Take-Two Interactive Software Inc.
Grand Theft Auto and Grand Theft Auto: San Andreas are registered trademarks of Take-Two Interactive Software Inc.

ISAMPP is licensed under MIT License
http://opensource.org/licenses/mit-license

3   Installation

Copy contents of *include* folder to *"[SA-MP Server folder]/include"* and also to *"include"* folder for Pawno (by default *"[SA-MP Server folder]/pawno/include"*).

In your game mode file you can include ISAMPP header for all its contents:

```
#include <i_sampp>
```

or include header files as you wish separately:

```
#include <i_sampp/i_colorlist>
#include <i_sampp/i_iconids>
#include <i_sampp/i_locationids>
#include <i_sampp/i_pickupids>
#include <i_sampp/i_skinids>
#include <i_sampp/i_vehids>
#include <i_sampp/i_weaponids>
#include <i_sampp/i_weatherids>
```

If you wish to run included sandbox game mode, you have to add file testinc.pwn to your *"gamemodes"* folder and compile it.

4   Includes

Every ISAMPP include file starts with prefix *i_*. Please keep in mind that ISAMPP may not be the only include with this prefix.

colorlist - List of color definitions
iconids - List of icon identifiers
locationids - List of location definitions with coordinates, interior identifiers and names
pickupids - List of definitions for pickup identifiers
skinids - List of definitions for character skin/model identifiers
vehids - List of definitions for all available vehicles
weaponids - List of definitions for weapon identifiers, sorted by weapon slot IDs
weatherids - List of definitions for weather identifiers

5   Color List

Color List contains definitions of some few common colors in two formats. Primary or Main color definitions have prefix *COLOR_* and are defined in 0xRRGGBBAA format. Secondary or String color definitions with prefix *SCOL_* are defined in more common hex format "{RRGGBB}" without the alpha color transparency values. The difference of these two formats can be explained by this example:

```
SendClientMessage(playerid, COLOR_RED, "Hello "SCOL_BLUE"World");
```

where the output will print in game client message box string of text in this format: **Hello** **World**

Secondary or String colors can be used simply in strings while Primary/Main colors would be recommended to be used as parameters. To see how every color looks in game, use in included sandbox mode for ISAMPP command */defcols[0-14]* or */stringcols*. Screenshot of color strings printed out in client message box is located in *"docs/i_colorlist.png"*.

6   Icon Identifiers

There are total 64 icons in total included as macro defines with prefix *ICON_*.

7   Location Identifiers

Locations, in-game interiors and exteriors, are stored in file *i_locationids.inc* as set of macro definitions for identifiers and three arrays. Array *locCoords* is storing float values of XYZ position on the map and float value of PlayerFacingAngle. By default, your PlayerInterior value is set to 0 whenever you are "outside" on the map. When player enters the interior, in order to load the interior you have to change PlayerInterior value accordingly. This is what array *locID* includes. Third array, *locName*, has defined strings of names of particular locations so this information can be displayed to player if needed.

Please note that if you wish to add new location to this list, you have to do it in correct order. Not every location has hitboxes because it was used only in cutscene, so in certain interiors you may just fall through ground.

8   Pickup Identifiers

Pickup identifiers are included as macro defines with prefix *PICKUP_*. Every pickup is placed on map in ISAMPP sandbox game mode.

9   Skin/Playermodel Identifiers

Skins are included as macro defines with prefix *SKIN_*, followed by name of model in game. Luckily, Rockstar North did great job in naming their models, so it's pretty simple to know what name represents which skin in game. Usually, with exception of few character models *(mainly main story ones and special characters)* where the name is obvious *(f.e. emmet = Emmet, etc.)*, character models are named in this or similar format: prefix *(f.e. if is located only in certain area)*, race, gender, age, suffix. In this context BMYRI = **B**lack, **M**ale, **Y**oung, **RI**ch. SBFYS = (**S**an **F**ransisco) **B**lack, **F**emale, **Y**oung, **S**treet, etc. This format is very helpful in contrast with number identifiers SA-MP uses by default.

10   Vehicle Identifiers

Vehicle identifiers are included as macro defines with prefix *VEH_* followed by the name of vehicle in game with capital letters *(motorcycles, bicycles, boats, planes, helicopters, trailers, trains, etc. are included in this format as well)*. For better orientation in list, defines are sorted by vehicle type.

11 Weapon Identifiers

Weapon identifiers are included as macro defines with prefix *WEAP_*. In ISAMPP sandbox game mode weapon pickups are fully functional. Every weapon in game can be used only for certain weapon slot and weapon identifiers are ordered in file *i_weaponids.inc* by weapon slot number.

12 Weather Identifiers

Weather identifiers are included as macro defines with prefix *WEATHER_*. In ISAMPP sandbox game mode you can test weather settings with command **/w** [**ID**].

13   Custom Functions

ISAMPP uses various stock functions which may be useful in creating your custom game modes for SA-MP or simply for debugging purposes. These stock functions are located in *i_sampp.inc* file.

13.1   ISAMPP_TELEPORT(playerid, locationid):

Teleports player to desired location passed as parameter 'locationid'.

Example of implementation:

```
if (strcmp("/teleport barbershop", cmdtext, true, 20) == 0) {
  ISAMPP_TELEPORT(playerid, LOC_BARBERSHOP);
  return 1;
}
```

13.2   ISAMPP_TELEPORTEX(playerid, locationid):

Same as ISAMPP_TELEPORT, plus outputs location name to in-game chat window.

Example of implementation:

```
if (strcmp("/teleport barbershop", cmdtext, true, 20) == 0) {
    ISAMPP_TELEPORTEX(playerid, LOC_BARBERSHOP);
    return 1;
}
```

13.3   ISAMPP_SHOWPLAYERPOSITION(playerid):

Outputs current player location coordinates and interior identifier to in-game chat window.

Example of implementation:

```
if (strcmp("/showplayerpos", cmdtext, true, 15) == 0) {
    ISAMPP_SHOWPLAYERPOSITION(playerid);
    return 1;
}
```