

United States Social Security Validation Object Library

Version 2.0 for Perl , PHP, and C++

Quentin Sager Consulting
20429 Ring Neck Road
Altoona, FL 32702
USA

www.quentinsagerconsulting.com
support@quentinsagerconsulting.com

The United States Social Security Validation Library is an object oriented class library that provides methods to perform basic validation of a United States Social Security Number.

The library is available in the Perl, PHP, and C++ programming languages. All versions share a common definition and naming convention for the validation object therefore we use a single description for each method.

The Perl version is designed for and requires Perl version 5 or above. It includes one source file, *ssn.pm*. This is a simple Perl module that requires no specific installation. It may be used with any script by placing it in the same directory as your Perl script and create the object. The required high group and area assignment data are embedded within the `__DATA__` section of the module itself and requires no external data.

The PHP version is designed for and requires PHP version 4 or above. It includes one source file, *ssn.class.php* and two associated data files; *ssnarea.dat* and *ssngroup.dat*. To use the object simply include the class file in your particular script and create the object.

The C++ version includes two source files, *clSsn.h* and *clSsn.cpp* and two associated data files; *ssnarea.dat* and *ssngroup.dat*. This version is delivered as source only. You must include and compile it with your particular application. It uses no compiler specific values or setting and may be compiled with virtually any C++ compiler on any platform.

::clSsn ()

This method is used to instantiate and initialize the validation object. For the Perl version only, the object is instantiated by creating a new instance of the ssn module, which equates to the PHP and C++ clSsn object constructor.

Parameters:

none

Examples:

Perl

```
# we assume the ssn module is in the current directory
use ssn;

my $SsnObject = new ssn;
```

PHP

```
<?php
require_once("./ssn.class.php");

my $SsnObject = new clSsn;
?>
```

C++

```
#include "clSsn.h"

clSsn SsnObject;           // static allocation
clSsn *SsnObject = new clSsn; // dynamic allocation
```

::Get()

Returns the 9 digit Social Security Number processed by the last validation without delimiters.

Parameters:

none

Returns:

Perl

PHP

String copy of the validated Social Security Number.

C++

const char * to the nul terminated validated Social Security Number.

Examples:

Perl

```
# we assume the ssn module is in the current directory
use ssn;

my $SsnObject = new ssn;

# if valid number display it without delimiters
if ($SsnObject->IsValid('123-45-6789'))
{
    print $SsnObject->Get();
}
```

PHP

```
<?php
require_once("./ssn.class.php");

my $SsnObject = new cISsn;

// if valid number display it without delimiters
if ($SsnObject->IsValid('123-45-6789'))
    echo $SsnObject->Get();
?>
```

C++

```
#include "cISsn.h"

cISsn *SsnObject = new cISsn;    // dynamic allocation

// if valid number display it without delimiters
if (SsnObject->IsValid("123-45-6789"))
    printf(SsnObject->Get());
```

::GetArea()
::GetGroup()
::GetSerial()

A United States Social Security Number is nine digits long and is composed of three parts. The first three digits are the *Area Number*, the following two digits are the *Group Number*, and the last four digits are the *Serial Number*. These methods are used to retrieve these components from the last number validated.

Parameters:

none

Returns:

Perl

PHP

String copy of the validated Social Security Number component.

C++

const char * to the nul terminated validated Social Security Number component.

Examples:

Perl

```
# we assume the ssn module is in the current directory
use ssn;

my $SsnObject = new ssn;

# if valid number display with formatting
if ($SsnObject->IsValid('123-45-6789')
    {
    printf("%s-%s-%s",$SsnObject->GetArea(),$SsnObject->GetGroup(),$SsnObject->GetSerial());
    }
```

PHP

```
<?php
require_once("./ssn.class.php");

my $SsnObject = new cISsn;

// if valid number display with formatting
if ($SsnObject->IsValid('123-45-6789')
    printf("%s-%s-%s",$SsnObject->GetArea(),$SsnObject->GetGroup(),$SsnObject->GetSerial());
?>
```

C++

```
#include "cISsn.h"

cISsn *SsnObject = new cISsn;    // dynamic allocation

// if valid number display with formatting
if (SsnObject->IsValid("123-45-6789")
    printf("%s-%s-%s",SsnObject->GetArea(),SsnObject->GetGroup(),SsnObject->GetSerial());
```

::GetAreaName()

This method is used to return the service or issuance area for the previously validated Social Security Number. The method retrieves the name by referencing the area portion of the Social Security Number against the assigned area data. For the PHP and C++ versions this data is contained in the external file *ssnarea.dat*. For the Perl version this data is embedded within the Perl Module itself.

Parameters:

none

Returns:

Perl
PHP

String copy of the validated Social Security Number issuance area.

C++

const char * to the nul terminated validated Social Security Number issuance area.

Examples:

Perl

```
# we assume the ssn module is in the current directory
use ssn;

my $SsnObject = new ssn;

# if valid number display with formatting
if ($SsnObject->IsValid('123-45-6789'))
{
    printf("%s-%s-%s\n", $SsnObject->GetArea(), $SsnObject->GetGroup(), $SsnObject->GetSerial());
    print $SsnObject->GetAreaName();
}
```

PHP

```
<?php
require_once("./ssn.class.php");

my $SsnObject = new clSsn;

// if valid number display with formatting
if ($SsnObject->IsValid('123-45-6789'))
{
    printf("%s-%s-%s\n", $SsnObject->GetArea(), $SsnObject->GetGroup(), $SsnObject->GetSerial());
    print $SsnObject->GetAreaName();
}
?>
```

C++

```
#include "clSsn.h"

clSsn *SsnObject = new clSsn;    // dynamic allocation

// if valid number display with formatting
if (SsnObject->IsValid("123-45-6789"))
{
    printf("%s-%s-%s\n", SsnObject->GetArea(), SsnObject->GetGroup(), SsnObject->GetSerial());
    printf(SsnObject->GetAreaName());
}
```

::IsValid(number)

Validates a Social Security Number. The passed number may contain strictly numeric values and /or numeric with common delimiters. The first test is the simple identification of invalid characters followed by a simple length test. Next the number is tested to ensure no single component contains an all zero value. Finally the area and group components are used to validate the group has been assigned for use within the area using the SSA group allocation algorithm.

The high group assignment test is made using the external data ssngrp.dat. For the Perl version only this data is embedded within the Perl Module.

Parameters:

number

Social Security Number to validate.

Returns:

- 0 - the number passed validation
- 1 - the number contains invalid characters
- 2 - the number length is incorrect
- 5 - the number's area section is invalid
- 6 - the number's group section is invalid
- 7 - the number's serial number section is invalid
- 8 - the number is reserved and not assignable
- 9 - the number's area has not been assigned