```
package HamScrape;

use diagnostics;
use strict;

use FileHandle;
use HTML::TableExtract;
use HTTP::Request::Common qw(GET POST);
use LWP::Simple;
use LWP::UserAgent;
use Net::HTTP;
use XML::Element;
use XML::TreeBuilder;

sub AppendElement
{
        my $tree = shift;
        my $elementName = shift;
        my $elementContent = shift;

    my $element = XML::Element->new($elementName);
    $element->push_content($elementContent);
    $tree->push_content($element);

        return $tree;
}

sub CreateStationXml(\%)
{
        my %argumentHash = %{(shift)};
        my @now = localtime;
        my $date = $now[4] . '/' . $now[3];

        my $tree = XML::Element->new('station', 'local-date' => $date);

        foreach my $elementName (keys (%argumentHash))
        {
                &AppendElement($tree, $elementName, $argumentHash{$elementName});
        }

    return $tree;
}

sub WriteRawTextToFile
{
        my %stationInfo = shift;
        my $callsign = $stationInfo{Callsign};
    my $timestamp =$stationInfo{LastHeard};
    my $position = $stationInfo{Position};
    my $status = $stationInfo{Status};

    my $output = new IO::File("station.txt", O_WRONLY);

    print $output "Callsign: $callsign\n";
    print $output "Timestamp: $timestamp\n";
    print $output "Position (Latitude & Longitude), Region: $position\n";
    print $output "Status: $status\n";
}

sub PrintXmlTree
{
        my $tree = shift;
        print STDOUT $tree->as_XML;
}
```

```
sub WriteXmlTreeToFile
{
        my $xmlTree = shift;
        my $handle = new IO::File ">station.xml" or die "Can't open station.xml for
 writing\n";
        print $handle $xmlTree;
}

sub FindU
{
        # Our requesting agent. Define our URL and POST.
        my $baseUrl  = 'http://www.findu.com/cgi-bin/find.cgi';
        my $post = "call=" . shift;

        # Here are the headers we'll send off...
        my $headers = HTTP::Headers->new(Accept => 'text/plain',
                        'User-Agent' => 'AutoLookup/1.0');

        # and the final requested web page.
        my $uable = HTTP::Request->new('POST', $baseUrl, $headers, $post);
        my $userAgent = LWP::UserAgent->new;
        my $request = $userAgent->request($uable);

        die $request->message unless $request->is_success;

        my $webPage = $request->content;

        $webPage =~ s/(<[^>]*>)*//isg; # Remove all HTML tags
        $webPage =~ s/ / /g; # As well as NBSPs

        my $callsign = "N/A";
        if ($webPage =~ m/Position of ([A-Z]{1,2}[0-9]{1}[A-Z]{2,3})/)
        {
                $callsign = $1;
        }
        # Write a regex that matches the following string:
        #       1.2 miles southwest of Bolingbrook, IL
        #       (\d)*.(\d)*\s(rest of string)

        my $status = "N/A";
        if ($webPage =~ m/Status: ([^,]*)/g)
        {
                $status = $1;
        }

        my $reportReceived = "N/A";
        if ($webPage =~ m/Report received  ([^\n]*)/)
        {
                $reportReceived = $1;
        }

        my $rawPacket = "N/A";
        if ($webPage =~ m/Raw packet: ([^\n]*)/)
        {
                $rawPacket = $1;
        }

        return ( Callsign=>$callsign, LastHeard=>$reportReceived, Status=>$status )
;
}

sub FindUXml
{
        my $queryCallsign = shift;
        my %aprsInfo = FindU($queryCallsign);
        my $xmlTree = CreateStationXml(%aprsInfo)->as_XML;
```

```
        return $xmlTree;
}

sub APRSWorld
{
        my $queryCallsign = shift;
        my $baseUrl = "http://db.aprsworld.net/datamart/switch.php?call=$queryCalls
ign&table=position&maps=yes";

        my $webPage = get("$baseUrl") or die $!;

        my $tableData = new HTML::TableExtract->new(headers => [
            "Callsign", "Date", "Latitude / Longitude", "Status"]);
        $tableData->parse($webPage);

        my $callsign;
        my $position;
        my $timestamp;
        my $status;

        foreach my $tableStates ($tableData->table_states) {
            foreach my $row ($tableStates->rows) {

                my $currentCallsign = @$row[0];
                next if length $currentCallsign == 0;
                $callsign = $currentCallsign;

                $timestamp = "Timestamp not available";
                if (defined(@$row[1])) {
                        $timestamp = @$row[1];
                }

                $position = "Position not available";
                if (defined(@$row[2])) {
                        $position = @$row[2];
                }

                $status = "Status not available";
                if (defined(@$row[4])) {
                        $status = @$row[4];
                }
            }
        }

        return ( Callsign=>$callsign, Position=>$position, LastHeard=>$timestamp, S
tatus=>$status );
}

sub APRSWorldXml
{
        my $queryCallsign = shift;
        my %aprsInfo = APRSWorld($queryCallsign);
        my $xmlTree = CreateStationXml(%aprsInfo)->as_XML;
        return $xmlTree;
}

sub QRZ
{
        my $queryCallsign = shift;

        my $baseUrl = "http://www.qrz.com/detail/";

        my $webPage = get("$baseUrl$queryCallsign") or die $!;
        $webPage =~ s/(<[^>]*>)*//isg;
        $webPage =~ s/ / /g;
```

```
        $webPage =~ m/Name:([^\n]*)/;
        my $name = $1;

        # This pattern currently does not match!
        $webPage =~ m/Class: (Novice|Technician|General)/;
        my $class = $1;

        $webPage =~ m/GMT Offset:([^\n]*)/;
        my $gmtOffset = $1;

        $webPage =~ m/Time Zone:([^\n]*)/;
        my $timeZone = $1;

        $webPage =~ m/Addr2:([^\n]*)/;
        my $cityStateZip = $1;

        $webPage =~ m/County:([^\n]*)/;
        my $county = $1;

        $webPage =~ m/Grid:([^\n]*)/;
        my $grid = $1;

        return ( Name=>$name, Class=>$class, CityStateZip => $cityStateZip, GMTOffs
et=>$gmtOffset, TimeZone=>$timeZone, Grid=>$grid );
}

sub QRZxml
{
        my $queryCallsign = shift;
        my %qrzInfo = QRZ($queryCallsign);
        my $xmlTree = CreateStationXml(%qrzInfo)->as_XML;
        return $xmlTree;
}

1;
```