
The Greenhouse Development Rights Calculator Web API

GDRs Calculator Team [mailto:calculator@gdrights.org]

January 2012

Table of Contents

Introduction	1
GET	1
Create a new database	1
Get parameter values	2
Get the valid range of years	2
Get general information about the calculator	2
POST	4
Error Messages	5
An Example Using PHP	5

Introduction

The Greenhouse Development Rights (GDRs) web API allows web applications to run the GDRs calculator and capture the outputs without having to install any code or display the GDRs calculator page. To use the API you must be familiar with web programming.

The GDRs web API uses **GET** and **POST**. To access the API, point to `http://gdrights.org/calculator/api`. The basic steps are either:

1. **GET** a database to use for a session, set user-specific parameter values using **POST**, and then repeatedly use **POST** to get results, or
2. Use **POST**, with no database, which will create a temporary database on each call.

The first version is much faster for repeated queries than the second one, especially if there is no need to change the parameter values after they first set.

Except for error messages and the response to the HTTP **OPTIONS** command (which simply says that **GET** and **POST** are the only options), all responses are JSON. Errors and the response to the **OPTIONS** command are plain text. If there is no error then the header is sent with the status 200 OK. Any other status signals an error.

GET

Use **GET** to either create a new database and get its location or to query parameter values. Here are the available commands, with sample responses.

Create a new database

```
GET [q=new_db]
```

```
prompt% curl --get http://gdrights.org/calculator/api/?q=new_db
```

```
{ "db" : "fw-sql3-DvaS6N" }
```

Get parameter values

The parameter values are set using the **POST** command. If the user changes parameter values, then the new values are stored in the user's copy of the database. This can be done anonymously, in which case the parameter values cannot be recovered. But if a database name is obtained using **q=new_db**, then the values are persistent, and can be recovered using **GET**.

Default values

The default parameter values apply unless they are overridden using **POST**.

```
GET [q=params]
```

```
prompt% curl --get http://gdrights.org/calculator/api/?q=params
```

```
{ "cum_since_yr": { "description": "The year when historical responsibility begins",  
"advanced": false, "db_param": "cumsince", "value": 1990, "min": 1850, "max": 2010, "step":  
:10, "list": null, "type": "int" }, "use_lulucf": { "description": "Include land-use emis  
sions in baseline (from 1950 only)", "advanced": false, "db_param": "use_lulucf", "va  
lue": 0, "min": 0, "max": 1, "step": 1, "list": null, "type": "int" }, "use_netexports": { "des  
cription": "Include emissions embodied in traded goods", "advanced": false, "db_para  
m": "use_netexports", "value": 0, "min": 0, "max": 1, "step": 1, "list": null, "type": "int" }  
, "use_nonco2": { "description": "Include non-CO2 gases in baseline (from 1990 only)  
...  
...
```

User-specified values

User-specified values are stored in the user's copy of the database.

```
GET [q=params] [db=dbname]
```

```
prompt% curl --get "http://gdrights.org/calculator/api/?q=params&db=fw-sql3-DvaS6N"
```

```
{ "cum_since_yr": { "description": "The year when historical responsibility begins",  
"advanced": false, "db_param": "cumsince", "value": 1900, "min": 1850, "max": 2010, "step":  
:10, "list": null, "type": "int" }, "use_lulucf": { "description": "Include land-use emis  
sions in baseline (from 1950 only)", "advanced": false, "db_param": "use_lulucf", "va  
lue": 0, "min": 0, "max": 1, "step": 1, "list": null, "type": "int" }, "use_netexports": { "des  
cription": "Include emissions embodied in traded goods", "advanced": false, "db_para  
m": "use_netexports", "value": 0, "min": 0, "max": 1, "step": 1, "list": null, "type": "int" }  
...  
...
```

Get the valid range of years

Depending on the baseline and emergency pathway, the valid range of years might be longer or shorter. To get the valid range of years for either the default or user-specific settings, use

```
GET [q=year_range] [db=dbname]
```

```
prompt% curl --get http://gdrights.org/calculator/api/?q=year_range
```

```
{ "min_year": "1990", "max_year": "2030" }
```

```
prompt% curl --get "http://gdrights.org/calculator/api/?q=year_range&db=fw-sql3-DvaS6N"
```

```
{ "min_year": "1850", "max_year": "2050" }
```

Get general information about the calculator

The calculator has general information that cannot be changed by the user. Consequently these should normally be called without specifying a database. However, as indicated for each command, it is possible to specify a database, which can be useful for debugging or to ensure that a default database has not been updated.

List of available pathways

When running the calculator, it is possible to pass one of a number of emergency pathways. The list is provided using **q=params**, but only by display name in order of ID number. This call provides an alternative way to get the list that returns the ID number, short code, and display name. When setting the *emergency_path* parameter, use the ID number.

```
GET [q=pathways] [db=dbname]
```

```
prompt% curl --get http://gdrights.org/calculator/api/?q=pathways
```

```
[{"id": "0", "short_code": "cum_750GtCO2", "display_name": "750 GtCO2 cumulative"}, {"id": "1", "short_code": "cum_1000GtCO2", "display_name": "1000 GtCO2 cumulative"}, {"id": "2", "short_code": "Energy_Rev", "display_name": "Energy Revolution"}, {"id": "3", "short_code": "Adv_Energy_Rev", "display_name": "Advanced Energy Revolution"}, {"id": "4", "short_code": "Hansen", "display_name": "Hansen-compliant pathway"}, {"id": "5", "short_code": "G8", "display_name": "G8 pathway"}, {"id": "6", "short_code": "GDRs_experimental", "display_name": "GDRs Experimental"}, {"id": "7", "short_code": "AOSIS", "display_name": "AOSIS"}, {"id": "8", "short_code": "IPCC_likely", "display_name": "IPCC \\\nLikely"}, {"id": "9", "short_code": "CAN", "display_name": "CAN"}]
```

Version number for the database or calculator

User databases expire if they are not used for two weeks. Therefore it is possible to maintain a user database over a very long time. It is good practice to check, at the start of each session, that the default and user database version numbers are the same. This can be done by using **q=data_ver** with and without specifying a database.

```
GET [q=calc_ver] [db=dbname]
```

```
GET [q=data_ver] [db=dbname]
```

```
prompt% curl --get http://gdrights.org/calculator/api/?q=calc_ver
```

```
"1.0.0"
```

```
prompt% curl --get "http://gdrights.org/calculator/api/?q=data_ver&db=fw-sql3-DvaS6N"
```

```
"5.3.4"
```

ISO 3-letter country codes recognized by the calculator

The calculator API reports national emissions labeled by almost-standard ISO 3-letter country codes. One notable difference is China, where the calculator combines data for mainland China (CHN) and Hong Kong (HKG); the combination is labeled CHK. Because the calculator codes are not fully standard, and because the standard itself has changed over time, it can be useful to check the definitions used by the calculator and API. Also, for presentation it is useful to have the country names corresponding to the codes. The **q=countries** query provides the list.

```
GET [q=countries] [db=dbname]
```

```
prompt% curl --get http://gdrights.org/calculator/api/?q=countries
```

```
[{"iso3": "ALB", "name": "Albania"}, {"iso3": "DZA", "name": "Algeria"}, {"iso3": "AGO", "name": "Angola"}, {"iso3": "ATG", "name": "Antigua and Barbuda"}, {"iso3": "ARG", "name": "Argentina"}, {"iso3": "ARM", "name": "Armenia"}, {"iso3": "AUS", "name": "Australia"}, {"iso3": "AUT", "name": "Austria"}, {"iso3": "AZE", "name": "Azerbaijan"}, {"iso3": "BHS", "name": "Bahamas, The"}, {"iso3": "BHR", "name": "Bahrain"}, {"iso3": "BGD", "name": "Bangladesh"}, {"iso3": "BRB", "name": "Barbados"}, {"iso3": "BLR", "name": "Belarus"}, {"iso3": "BEL", "name": "Belgium"}, {"iso3": "BLZ", "name": "Belize"}, {"iso3": "BEN", "name": "Benin"}, {"iso3": "BTN", "name": "Bhutan"}, {"iso3": "BOL", "name": "Bolivia"}, {"iso3": "BIH", "name": "Bosnia and Herzegovina"}, {"iso3": "BWA", "name": "Botswana"}, {"iso3": "BRA", "name": "Brazil"}, {"iso3": "BVT", "name": "Bouvet Island"}, {"iso3": "CAN", "name": "Canada"}, {"iso3": "CCK", "name": "Cocos (Keeling) Islands"}, {"iso3": "CHE", "name": "Switzerland"}, {"iso3": "CHL", "name": "Chile"}, {"iso3": "CHN", "name": "China"}, {"iso3": "CXR", "name": "Christmas Island"}, {"iso3": "CUB", "name": "Cuba"}, {"iso3": "CYM", "name": "Cayman Islands"}, {"iso3": "CYP", "name": "Cyprus"}, {"iso3": "DNK", "name": "Denmark"}, {"iso3": "DMA", "name": "Dominica"}, {"iso3": "DOM", "name": "Dominican Republic"}, {"iso3": "DUM", "name": "Dumont d'Urville Island"}, {"iso3": "ECU", "name": "Ecuador"}, {"iso3": "EGY", "name": "Egypt"}, {"iso3": "ERI", "name": "Eritrea"}, {"iso3": "ESH", "name": "Western Sahara"}, {"iso3": "FIN", "name": "Finland"}, {"iso3": "FJI", "name": "Fiji"}, {"iso3": "FLK", "name": "Falkland Islands (Malvinas)"}, {"iso3": "FRA", "name": "France"}, {"iso3": "GAB", "name": "Gabon"}, {"iso3": "GBR", "name": "United Kingdom"}, {"iso3": "GEO", "name": "Georgia"}, {"iso3": "GGY", "name": "Guernsey"}, {"iso3": "GHA", "name": "Ghana"}, {"iso3": "GIB", "name": "Gibraltar"}, {"iso3": "GLP", "name": "Guadeloupe"}, {"iso3": "GNB", "name": "Guinea-Bissau"}, {"iso3": "GRC", "name": "Greece"}, {"iso3": "GRD", "name": "Grenada"}, {"iso3": "GTM", "name": "Guatemala"}, {"iso3": "GUF", "name": "Guiana, French"}, {"iso3": "GUY", "name": "Guyana"}, {"iso3": "HKG", "name": "Hong Kong"}, {"iso3": "HND", "name": "Honduras"}, {"iso3": "HRV", "name": "Croatia"}, {"iso3": "HUN", "name": "Hungary"}, {"iso3": "IDN", "name": "Indonesia"}, {"iso3": "IND", "name": "India"}, {"iso3": "ISL", "name": "Iceland"}, {"iso3": "ITA", "name": "Italy"}, {"iso3": "JAM", "name": "Jamaica"}, {"iso3": "JPN", "name": "Japan"}, {"iso3": "JYU", "name": "Yugoslavia"}, {"iso3": "KAZ", "name": "Kazakhstan"}, {"iso3": "KHM", "name": "Cambodia"}, {"iso3": "KIR", "name": "Kiribati"}, {"iso3": "KOR", "name": "Korea"}, {"iso3": "KWT", "name": "Kuwait"}, {"iso3": "KGZ", "name": "Kyrgyzstan"}, {"iso3": "LAC", "name": "Laos"}, {"iso3": "LBN", "name": "Lebanon"}, {"iso3": "LBR", "name": "Liberia"}, {"iso3": "LCA", "name": "Saint Lucia"}, {"iso3": "LIE", "name": "Liechtenstein"}, {"iso3": "LIT", "name": "Lithuania"}, {"iso3": "LUX", "name": "Luxembourg"}, {"iso3": "LVA", "name": "Latvia"}, {"iso3": "MAC", "name": "Macao"}, {"iso3": "MDA", "name": "Moldova"}, {"iso3": "MEX", "name": "Mexico"}, {"iso3": "MHL", "name": "Marshall Islands"}, {"iso3": "MKD", "name": "Macedonia"}, {"iso3": "MLT", "name": "Malta"}, {"iso3": "MNI", "name": "Minnamoon Island"}, {"iso3": "MNP", "name": "Mariana Islands"}, {"iso3": "MOR", "name": "Morocco"}, {"iso3": "MOT", "name": "Motu Island"}, {"iso3": "MUS", "name": "Mauritius"}, {"iso3": "MYA", "name": "Myanmar"}, {"iso3": "NAM", "name": "Namibia"}, {"iso3": "NLD", "name": "Netherlands"}, {"iso3": "NOR", "name": "Norway"}, {"iso3": "NRU", "name": "Nauru"}, {"iso3": "NZL", "name": "New Zealand"}, {"iso3": "OMN", "name": "Oman"}, {"iso3": "PAK", "name": "Pakistan"}, {"iso3": "PAN", "name": "Panama"}, {"iso3": "PCN", "name": "Pitcairn Islands"}, {"iso3": "PER", "name": "Peru"}, {"iso3": "PHL", "name": "Philippines"}, {"iso3": "PLW", "name": "Palau"}, {"iso3": "POL", "name": "Poland"}, {"iso3": "PRI", "name": "Puerto Rico"}, {"iso3": "PRY", "name": "Paraguay"}, {"iso3": "PSE", "name": "Palestine"}, {"iso3": "PYF", "name": "French Polynesia"}, {"iso3": "PUS", "name": "Pitcairn Island"}, {"iso3": "QAT", "name": "Qatar"}, {"iso3": "ROU", "name": "Romania"}, {"iso3": "RUS", "name": "Russia"}, {"iso3": "RWA", "name": "Rwanda"}, {"iso3": "SAU", "name": "Saudi Arabia"}, {"iso3": "SBD", "name": "Solomon Islands"}, {"iso3": "SDN", "name": "Sudan"}, {"iso3": "SEN", "name": "Senegal"}, {"iso3": "SGP", "name": "Singapore"}, {"iso3": "SHN", "name": "Shanley Island"}, {"iso3": "SJM", "name": "Jan Mayen"}, {"iso3": "SLB", "name": "Slebolle Islands"}, {"iso3": "SLV", "name": "El Salvador"}, {"iso3": "SLO", "name": "Slovenia"}, {"iso3": "SMR", "name": "San Marino"}, {"iso3": "SOM", "name": "Somalia"}, {"iso3": "SRI", "name": "Sri Lanka"}, {"iso3": "STP", "name": "Sao Tome and Principe"}, {"iso3": "SUR", "name": "Suriname"}, {"iso3": "SVA", "name": "Sweden"}, {"iso3": "SWZ", "name": "Swaziland"}, {"iso3": "SWE", "name": "Sweden"}, {"iso3": "SXM", "name": "Sint Eustachius"}, {"iso3": "SYC", "name": "Seychelles"}, {"iso3": "SYR", "name": "Syria"}, {"iso3": "TAN", "name": "Tanzania"}, {"iso3": "TGO", "name": "Togo"}, {"iso3": "TKM", "name": "Turkmenistan"}, {"iso3": "TKL", "name": "Tokelau"}, {"iso3": "TLS", "name": "Timor-Leste"}, {"iso3": "TON", "name": "Tonga"}, {"iso3": "TUN", "name": "Tunisia"}, {"iso3": "TUR", "name": "Turkey"}, {"iso3": "TUV", "name": "Tuvalu"}, {"iso3": "TZA", "name": "Tanzania"}, {"iso3": "UGA", "name": "Uganda"}, {"iso3": "UKR", "name": "Ukraine"}, {"iso3": "URY", "name": "Uruguay"}, {"iso3": "USA", "name": "United States"}, {"iso3": "UZB", "name": "Uzbekistan"}, {"iso3": "VAT", "name": "Vatican"}, {"iso3": "VCT", "name": "Saint Vincent and the Grenadines"}, {"iso3": "VEN", "name": "Venezuela"}, {"iso3": "VUT", "name": "Vanuatu"}, {"iso3": "WLF", "name": "Wallis and Futuna"}, {"iso3": "WSM", "name": "Samoa"}, {"iso3": "YEM", "name": "Yemen"}, {"iso3": "ZMB", "name": "Zambia"}, {"iso3": "ZWE", "name": "Zimbabwe"}]
```

...

List of region names and codes used by the calculator

The regions defined by the calculator are for the most part commonly-used regions. However, unlike for countries there is no set of ISO standard region codes. The **q=regions** query provides the list of region codes and names that the calculator recognizes.

```
GET [q=regions] [db=dbname]
```

```
prompt% curl --get http://gdrights.org/calculator/api/?q=regions
```

```
[{"region_code": "high_income", "name": "High Income"}, {"region_code": "upper_mid_income", "name": "Upper Middle Income"}, {"region_code": "lower_mid_income", "name": "Lower Middle Income"}, {"region_code": "low_income", "name": "Low Income"}, {"region_code": "annex_1", "name": "Annex 1"}, {"region_code": "annex_2", "name": "Annex 2"}, {"region_code": "non_annex_1", "name": "Non-Annex 1"}, {"region_code": "eit", "name": "EITs"}, {"region_code": "ldc", "name": "LDCs"}, {"region_code": "eu15", "name": "EU 15"}, {"region_code": "eu12", "name": "EU 12+"}, {"region_code": "eu27", "name": "EU 27"}, {"region_code": "OECD_NA", "name": "OECD North America"}, {"region_code": "OECD", "name": "OECD"}, {"region_code": "OECD_Europe", "name": "OECD Europe"}, {"region_code": "OECD_Pacific", "name": "OECD Pacific"}, {"region_code": "EE_Eurasia", "name": "Eastern Europe and Eurasia"}, {"region_code": "Asia", "name": "Non-OECD Asia"}, {"region_code": "Africa", "name": "Africa"}, {"region_code": "Middle_East", "name": "Middle East"}, {"region_code": "Latin_America", "name": "Latin America"}, {"region_code": "Non-OECD", "name": "Non-OECD"}]
```

POST

There are several options for POST which allow you to set parameters and get results.

```
POST [reset=yes] [db=dbname] [years=yearlist] [countries=countrylist] [param1=val1] [...] [paramN=valN]
```

- The *yearlist* can be a single year, several years separated by commas, a range separated by colon, or a combination. For example, **years=1990:1995,2020,2030**.
- The *countrylist* is a comma-separated list of country ISO 3-letter codes or GDRs region codes. (You can get the list of recognized ISO 3-letter codes and region codes using **GET**, as described above.) For example, **countries=USA,BRA,annex_1,eu27**.

Here are some examples of using **POST**.

Example 1. Run with defaults and get all values

```
prompt% curl http://gdrights.org/calculator/api/ -d ""
```

```
[{"code": "AFG", "name": "Afghanistan", "year": "1990", "pop_mln": "12.580412", "gdp_bln_USDMER": "16.30566861", "gdp_bln_USDPPP": "28.1299980236345", "fossil_CO2_MtCO2": "6.319289999", "LULUCF_MtCO2": "5.418505675", "NonCO2_MtCO2e": "9.800000001", "vol_rdxn_MtCO2": "0.0", "gdrs_alloc_MtCO2": "6.319291", "gdrs_r_MtCO2": "0.175793676666667", "a1_dom_rdxn_MtCO2": "0.0", "net_import_MtCO2": "0.0", "gdrs_c_bln_USDMER": "0.4536006", "gdrs_rci": "2.167867e-05", "gdrs_pop_mln_above_dl": "0.2844022", "gdrs_pop_mln_above_lux": "8.127849e-05", "lux_emiss_MtCO2": "0.000144008993333333", "lux_emiss_applied_MtCO2": "0.0"}, {"code": "AFG", "name": "Afghanistan", "year": "1990", "pop_mln": "12.580412", "gdp_bln_USDMER": "16.30566861", "gdp_bln_USDPPP": "28.1299980236345", "fossil_CO2_MtCO2": "6.319289999", "LULUCF_MtCO2": "5.418505675", "NonCO2_MtCO2e": "9.800000001", "vol_rdxn_MtCO2": "0.0", "gdrs_alloc_MtCO2": "6.319291", "gdrs_r_MtCO2": "0.175793..."}]
```

Example 2. Run with defaults and get values only for Annex I in 2020

```
prompt% curl http://gdrights.org/calculator/api/ -d "years=2020&countries=annex_1"
```

```
[{"code":"annex_1","name":"Annex 1","year":"2020","pop_mln":"1328.3852969251","gdp_blnUSDMER":"44486.978644525","gdp_blnUSDPPP":"50369.294554809","fossil_CO2_MtCO2":"14545.8260134697","LULUCF_MtCO2":"-197.956074004333","NonCO2_MtCO2e":"2883.44116621","vol_rdxn_MtCO2":"0.002884570333333333","gdrs_alloc_MtCO2":"-2633.72171949333","gdrs_r_MtCO2":"253410.876811767","al_dom_rdxn_MtCO2":"0.0","net_import_MtCO2":"2134.9619323928","gdrs_c_blnUSDMER":"36723.8228065","gdrs_rci":"0.74380656614","gdrs_pop_mln_above_dl":"1221.93855634","gdrs_pop_mln_above_lux":"644.038437103","lux_emiss_MtCO2":"5737.14838759333","lux_emiss_applied_MtCO2":"5737.14838759333"}]
```

Example 3. Run with a specified database, change a parameter, and get values for Annex I in 2020

```
prompt% curl http://gdrights.org/calculator/api/ -d "db=fw-sql3-DvaS6N&cum_since_yr=1900&years=2020&countries=annex_1"
```

```
[{"code":"annex_1","name":"Annex 1","year":"2020","pop_mln":"1328.3852969251","gdp_blnUSDMER":"44486.978644525","gdp_blnUSDPPP":"50369.294554809","fossil_CO2_MtCO2":"14545.8260134697","LULUCF_MtCO2":"-197.956074004333","NonCO2_MtCO2e":"2883.44116621","vol_rdxn_MtCO2":"0.002884570333333333","gdrs_alloc_MtCO2":"-3369.55918592333","gdrs_r_MtCO2":"501603.6263854","al_dom_rdxn_MtCO2":"0.0","net_import_MtCO2":"2134.9619323928","gdrs_c_blnUSDMER":"36723.8228065","gdrs_rci":"0.791639306343","gdrs_pop_mln_above_dl":"1221.93855634","gdrs_pop_mln_above_lux":"644.038437103","lux_emiss_MtCO2":"5737.14838759333","lux_emiss_applied_MtCO2":"5737.14838759333"}]
```

Example 4. Run with a specified database and emergency pathway, and get values for Annex I in 2020

```
prompt% curl http://gdrights.org/calculator/api/ -d "db=fw-sql3-TbGUvI&emergency_path=2&years=2020&countries=annex_1"
```

```
[{"code":"annex_1","name":"Annex 1","year":"2020","pop_mln":"1328.3852969251","gdp_blnUSDMER":"44486.978644525","gdp_blnUSDPPP":"50369.294554809","fossil_CO2_MtCO2":"14545.8260134697","LULUCF_MtCO2":"-197.956074004333","NonCO2_MtCO2e":"2883.44116621","vol_rdxn_MtCO2":"0.002884570333333333","gdrs_alloc_MtCO2":"7900.698092903","gdrs_r_MtCO2":"351316.603934267","al_dom_rdxn_MtCO2":"0.0","net_import_MtCO2":"2134.9619323928","gdrs_c_blnUSDMER":"36723.8228065","gdrs_rci":"0.71066078396","gdrs_pop_mln_above_dl":"1221.93855634","gdrs_pop_mln_above_lux":"644.038437103","lux_emiss_MtCO2":"6491.60228086333","lux_emiss_applied_MtCO2":"6491.60228086333"}]
```

Error Messages

Only two error codes can be returned.

- 410 Gone: If the requested database named in a **db=dbname** parameter does not exist, a 410 code is returned, along with text saying the database no longer exists.
- 400 Bad request: (**GET** only): If the parameters passed to the API are invalid, a 400 code is returned, with text saying what the valid parameters are.

An Example Using PHP

The API can be called from any language that supports HTTP client functions, which means it can be used from almost any modern programming language. Also, there are several libraries and compiled extensions for PHP for making HTTP

transactions. For concreteness this example uses a specific PHP library, the PEAR HTTP_Request library. To keep the example as simple as possible the script does not feature some good coding practices that would make it more complex (although shorter). Specifically, there is redundant code and code that could be encapsulated in a class.

To see the example in action, go to http://gdrights.org/calculator_dev/api/example.php.

```
<?php
require_once "HTTP/Request.php";

function get_countries() {
    $req =& new HTTP_Request("http://gdrights.org/calculator/api/?q=countries");
    $req->setMethod(HTTP_REQUEST_METHOD_GET);
    if (!PEAR::isError($req->sendRequest())) {
        // Note: json_decode returns arrays as StdClass, so have to cast
        $response = (array) json_decode($req->getResponseBody());
    } else {
        throw new Exception($req->getMessage());
    }
    return $response;
}

function get_params() {
    $req =& new HTTP_Request("http://gdrights.org/calculator/api/?q=params");
    $req->setMethod(HTTP_REQUEST_METHOD_GET);
    if (!PEAR::isError($req->sendRequest())) {
        $response = (array) json_decode($req->getResponseBody());
    } else {
        throw new Exception($req->getMessage());
    }
    return $response;
}

function get_country_data($iso3) {
    $req =& new HTTP_Request("http://gdrights.org/calculator/api/");
    $req->setMethod(HTTP_REQUEST_METHOD_POST);
    $req->addPostData('years', 2020); // Note hard-coded year
    $req->addPostData('countries', $iso3);
    if (!PEAR::isError($req->sendRequest())) {
        $response = json_decode($req->getResponseBody());
        // The decode procedure duplicates the first element: get the tail
        $response = array_slice($response, 1);
    } else {
        throw new Exception($req->getMessage());
    }
    return $response;
}
?>

<html>
<head>
    <title>Testing GDRs API</title>
</head>
<body>
    <h1>Testing GDRs API</h1>
    <form method="post">
        <select name="country">
            <?php
                $countries = get_countries();
                foreach ($countries as $country) {
                    $ca = (array) $country;
```

```
        $option = '<option value=""';
        $option .= $ca['iso3'];
        $option .= '>';
        $option .= $ca['name'];
        $option .= '</option>';
        echo $option;
    }
    ?>
</select>
<input type="submit" value="calculate" />
</form>
<?php
    if (isset($_POST['country'])) {
        echo '<h2>output</h2>';
        echo '<table>';
        echo '<tr><td>item</td><td>value</td></tr>';
        $country_data_list = get_country_data($_POST['country']);
        $country_data = (array) $country_data_list[0];
        foreach ($country_data as $item=>$value) {
            $row = '<tr>';
            $row .= '<td>' . $item . '</td>';
            $row .= '<td>' . $value . '</td>';
            $row .= '</tr>';
            echo $row;
        }
        echo '</table>';

        echo '<h2>parameters</h2>';
        echo '<table>';
        echo '<tr><td>item</td><td>value</td></tr>';
        $params = get_params();
        foreach ($params as $item=>$info) {
            $info_array = (array) $info;
            $row = '<tr>';
            $row .= '<td>' . $item . '</td>';
            $row .= '<td>' . $info_array['value'] . '</td>';
            $row .= '</tr>';
            echo $row;
        }
        echo '</table>';
    }
    ?>
</body>
</html>
```