Reading data

In this part we'll get the crisis years from a published IMF data set.

First let's get the standard country codes coresponding to these countries using dracula.

In [12]:
```
import inspect
from dracula.extractor import Extractor
import dracula
extractor = Extractor()
country_codes = {}
countries = extractor.grab_metadata("countries")
print(inspect.getsourcelines(dracula.wb.parser.parse_multiple_countries_alone))
for country in countries:
  #print(dir(country))
  country_codes[country.name]=country.code
print(country_codes)
```

```
(['def parse_multiple_countries_alone(data_list):\n', '    """\n', '    Parse a query for
countries without indicators\n', '    @return: list of countries\n', '    """\n', "
get_country_id = lambda item : item['id']\n", "    get_country_id_iso2 = lambda item :
item['iso2Code']\n", '    countries = []\n', '    data_list.sort(key = get_country_id)\n', '
for item in data_list:\n', '        current_id = get_country_id(item)\n', '        current_id_iso2
= get_country_id_iso2(item)\n', '        country = Country(current_id)\n', '
country.code_iso2 = current_id_iso2\n', "        country.name = item['name']\n", '
countries.append(country)\n', '    return countries\n'], 63)
{u'Canada': u'CAN', u'Sao Tome and Principe': u'STP', u'Turkmenistan': u'TKM', u'Lao
PDR': u'LAO', u'Arab World': u'ARB', u'Latin America & Caribbean (all income levels)':
u'LCN', u'Cambodia': u'KHM', u'Ethiopia': u'ETH', u'Aruba': u'ABW', u'Swaziland': u'SWZ',
u'South Asia': u'SAS', u'Argentina': u'ARG', u'Bolivia': u'BOL', u'Bahamas, The': u'BHS',
u'Burkina Faso': u'BFA', u'OECD members': u'OED', u'Ghana': u'GHA', u'Saudi Arabia':
u'SAU', u'Rwanda': u'RWA', u'Japan': u'JPN', u'Channel Islands': u'CHI', u'American
Samoa': u'ASM', u'Northern Mariana Islands': u'MNP', u'Slovenia': u'SVN', u'Guatemala':
u'GTM', u'Bosnia and Herzegovina': u'BIH', u'Kuwait': u'KWT', u'Russian Federation':
u'RUS', u'Jordan': u'JOR', u'St. Lucia': u'LCA', u'Congo, Rep.': u'COG', u'Dominica':
u'DMA', u'Liberia': u'LBR', u'Maldives': u'MDV', u'East Asia & Pacific (all income levels)':
u'EAS', u'Virgin Islands (U.S.)': u'VIR', u'Lithuania': u'LTU', u'Tanzania': u'TZA', u'Vietnam':
u'VNM', u'Albania': u'ALB', u'Gabon': u'GAB', u'Monaco': u'MCO', u'New Zealand': u'NZL',
u'European Union': u'EUU', u'Jamaica': u'JAM', u'Greenland': u'GRL', u'Samoa': u'WSM',
u'Slovak Republic': u'SVK', u'United Arab Emirates': u'ARE', u'Guam': u'GUM',
u'Uruguay': u'URY', u'India': u'IND', u'Azerbaijan': u'AZE', u'Lesotho': u'LSO', u'Kenya':
u'KEN', u'Upper middle income': u'UMC', u'Tajikistan': u'TJK', u'Pacific island small
states': u'PSS', u'Turkey': u'TUR', u'Afghanistan': u'AFG', u'Venezuela, RB': u'VEN',
u'Bangladesh': u'BGD', u'Mauritania': u'MRT', u'Solomon Islands': u'SLB', u'Korea, Rep.':
u'KOR', u'San Marino': u'SMR', u'Mongolia': u'MNG', u'France': u'FRA', u'Syrian Arab
Republic': u'SYR', u'Bermuda': u'BMU', u'Namibia': u'NAM', u'Somalia': u'SOM', u'Peru':
u'PER', u'Vanuatu': u'VUT', u'Nigeria': u'NGA', u'Seychelles': u'SYC', u'Norway': u'NOR',
u"Cote d'Ivoire": u'CIV', u'Europe & Central Asia (developing only)': u'ECA', u'Benin':
u'BEN', u'Other small states': u'OSS', u'Cuba': u'CUB', u'Cameroon': u'CMR',
u'Montenegro': u'MNE', u'Low & middle income': u'LMY', u'Togo': u'TGO', u'China':
u'CHN', u'Sub-Saharan Africa (developing only)': u'SSA', u'Armenia': u'ARM', u'Small
states': u'SST', u'Timor-Leste': u'TLS', u'Dominican Republic': u'DOM', u'Low income':
```

u'LIC', u'Ukraine': u'UKR', u'Bahrain': u'BHR', u'Tonga': u'TON', u'Finland': u'FIN', u'Latin America & Caribbean (developing only)': u'LAC', u'High income': u'HIC', u'Libya': u'LBY', u'Cayman Islands': u'CYM', u'Central African Republic': u'CAF', u'Europe & Central Asia (all income levels)': u'ECS', u'Mauritius': u'MUS', u'Liechtenstein': u'LIE', u'Belarus': u'BLR', u'Mali': u'MLI', u'Micronesia, Fed. Sts.': u'FSM', u'Korea, Dem. Rep.': u'PRK', u'Bulgaria': u'BGR', u'North America': u'NAC', u'Romania': u'ROU', u'Angola': u'AGO', u'Egypt, Arab Rep.': u'EGY', u'Trinidad and Tobago': u'TTO', u'St. Vincent and the Grenadines': u'VCT', u'Cyprus': u'CYP', u'Caribbean small states': u'CSS', u'Brunei Darussalam': u'BRN', u'Qatar': u'QAT', u'Middle income': u'MIC', u'Austria': u'AUT', u'High income: OECD': u'OEC', u'Mozambique': u'MOZ', u'Uganda': u'UGA', u'Kyrgyz Republic': u'KGZ', u'Hungary': u'HUN', u'Niger': u'NER', u'United States': u'USA', u'Brazil': u'BRA', u'World': u'WLD', u'Middle East & North Africa (all income levels)': u'MEA', u'Guinea': u'GIN', u'Panama': u'PAN', u'Costa Rica': u'CRI', u'Luxembourg': u'LUX', u'Cape Verde': u'CPV', u'Andorra': u'AND', u'Chad': u'TCD', u'Euro area': u'EMU', u'Ireland': u'IRL', u'Pakistan': u'PAK', u'Palau': u'PLW', u'Faeroe Islands': u'FRO', u'Lower middle income': u'LMC', u'Ecuador': u'ECU', u'Czech Republic': u'CZE', u'Australia': u'AUS', u'Algeria': u'DZA', u'El Salvador': u'SLV', u'Tuvalu': u'TUV', u'St. Kitts and Nevis': u'KNA', u'Marshall Islands': u'MHL', u'Chile': u'CHL', u'Puerto Rico': u'PRI', u'Belgium': u'BEL', u'Kiribati': u'KIR', u'Haiti': u'HTI', u'Belize': u'BLZ', u'Sierra Leone': u'SLE', u'Georgia': u'GEO', u'East Asia & Pacific (developing only)': u'EAP', u'Denmark': u'DNK', u'Philippines': u'PHL', u'Moldova': u'MDA', u'Macedonia, FYR': u'MKD', u'Morocco': u'MAR', u'Croatia': u'HRV', u'French Polynesia': u'PYF', u'Guinea-Bissau': u'GNB', u'Thailand': u'THA', u'Switzerland': u'CHE', u'Grenada': u'GRD', u'Yemen, Rep.': u'YEM', u'Isle of Man': u'IMN', u'Portugal': u'PRT', u'Estonia': u'EST', u'Kosovo': u'KSV', u'Sweden': u'SWE', u'Mexico': u'MEX', u'Hong Kong SAR, China': u'HKG', u'South Africa': u'ZAF', u'Uzbekistan': u'UZB', u'Djibouti': u'DJI', u'West Bank and Gaza': u'PSE', u'Antigua and Barbuda': u'ATG', u'Spain': u'ESP', u'Colombia': u'COL', u'Burundi': u'BDI', u'Least developed countries: UN classification': u'LDC', u'Fiji': u'FJI', u'Barbados': u'BRB', u'Madagascar': u'MDG', u'Italy': u'ITA', u'Curacao': u'CUW', u'Bhutan': u'BTN', u'Sudan': u'SDN', u'Nepal': u'NPL', u'Singapore': u'SGP', u'Malta': u'MLT', u'Netherlands': u'NLD', u'Macao SAR, China': u'MAC', u'Suriname': u'SUR', u'Middle East & North Africa (developing only)': u'MNA', u'Turks and Caicos Islands': u'TCA', u'St. Martin (French part)': u'MAF', u'Iran, Islamic Rep.': u'IRN', u'Israel': u'ISR', u'Indonesia': u'IDN', u'Malaysia': u'MYS', u'Iceland': u'ISL', u'Zambia': u'ZMB', u'Sub-Saharan Africa (all income levels)': u'SSF', u'Senegal': u'SEN', u'Papua New Guinea': u'PNG', u'Malawi': u'MWI', u'Zimbabwe': u'ZWE', u'Germany': u'DEU', u'Oman': u'OMN', u'Kazakhstan': u'KAZ', u'Poland': u'POL', u'Sint Maarten (Dutch part)': u'SXM', u'Eritrea': u'ERI', u'Iraq': u'IRQ', u'New Caledonia': u'NCL', u'Paraguay': u'PRY', u'Not classified': u'INX', u'Latvia': u'LVA', u'South Sudan': u'SSD', u'Guyana': u'GUY', u'Honduras': u'HND', u'Myanmar': u'MMR', u'Equatorial Guinea': u'GNQ', u'Tunisia': u'TUN', u'Nicaragua': u'NIC', u'Congo, Dem. Rep.': u'COD', u'Serbia': u'SRB', u'Botswana': u'BWA', u'United Kingdom': u'GBR', u'Gambia, The': u'GMB', u'High income: nonOECD': u'NOC', u'Greece': u'GRC', u'Sri Lanka': u'LKA', u'Lebanon': u'LBN', u'Comoros': u'COM', u'Heavily indebted poor countries (HIPC)': u'HPC'}

Manual fixing

In [13]:
```
country_codes["Serbia, Republic of"] = 'SRB'
country_codes['Brunei'] = 'BRN'
country_codes[u"Lao People's Dem. Rep."] = 'LAO'
country_codes['Venezuela'] = 'VEN'
country_codes['Korea'] = 'KOR'
country_codes['Luxemburg'] = 'LUX'
country_codes['Congo, Dem. Rep. of'] = 'COG'
country_codes['Central African Rep.'] = 'CAF'
country_codes['China, P.R.: Hong Kong'] = 'HKG'
```

```
country_codes[u'Côte d'Ivoire'] = 'CIV'
country_codes['China, P.R.'] = 'CHN'
country_codes['Macedonia'] = 'MKD'
country_codes['Congo, Rep. of'] = 'COG'
country_codes['Iran, I.R. of'] = 'IRN'
country_codes['Egypt'] = 'EGY'
country_codes[u'São Tomé and Principe'] = 'STP'
country_codes['Yemen'] = 'YEM'
country_codes['Venezuela'] = 'VEN'
country_codes['Russia'] = 'RUS'
country_codes['Luxemburg'] = 'LUX'
country_codes['Gibraltar'] = 'GIB'
```

Now, let's get busy reading the IMF data into dictionaries with country codes as keys and years as values.

In [14]:
```
import xlrd
def add_crisis(what, where, key):
    #where[key].add(what)
    if what!=None:
        try:
            where[key].add(what)
        except KeyError:
            where[key] = set([what])

def parse_imf_db(loc):
    wb = xlrd.open_workbook(loc)
    sh = wb.sheet_by_index(0)
    banking_crises = {}
    currency_crises = {}
    debt_crises = {}
    problematic = []
    for rownum in range(3,sh.nrows-1):
        country = sh.row_values(rownum)[0].rstrip()
        try:
            code = country_codes[country]
        except KeyError:
            code = country # we'll fix those maunally
            problematic.append(code)
        years = [int(x) if x!="" else None for x in sh.row_values(rownum)[1:]]
        add_crisis(years[0], banking_crises, code)
        add_crisis(years[1], currency_crises, code)
        add_crisis(years[2], debt_crises, code)
        #print(country, years)
    # Test to se there are no problems (should be empty).
    print("Problematic:")
    print(problematic)
    return banking_crises, currency_crises, debt_crises
import os
loc = os.path.expanduser("~/Dropbox/dev/itd/skripte/ipy_notebook/data/imf/IMF Finan
banking_crises, currency_crises, debt_crises = parse_imf_db(loc)
print(currency_crises)
```

```
Problematic:
[]
```

```
{u'DZA': set([1994, 1988]), u'AGO': set([1996, 1991]), 'EGY': set([1979, 1990]), u'BGD':
set([1976]), u'NAM': set([1984]), u'BGR': set([1996]), u'BOL': set([1973, 1981]), u'GHA':
set([2000, 1993, 1978, 1983]), u'PAK': set([1972]), u'JOR': set([1989]), u'LBY':
set([2002]), u'MYS': set([1998]), u'TZA': set([1985, 1990]), u'PRT': set([1983]), u'KHM':
set([1992, 1971]), u'PRY': set([1984, 2002, 1989]), u'LBN': set([1984, 1990]), u'BFA':
set([1994]), u'MRT': set([1993]), u'CHL': set([1972, 1982]), u'JAM': set([1978, 1991,
1983]), u'GIN': set([2005, 1982]), u'FIN': set([1993]), u'URY': set([2002, 1972, 1990,
1983]), u'THA': set([1998]), u'NPL': set([1984, 1992]), u'MAR': set([1981]), 'YEM':
set([1985, 1995]), u'PHL': set([1998, 1983]), u'ZAF': set([1984]), u'NIC': set([1985,
1979, 1990]), u'TGO': set([1994]), u'SYR': set([1988]), u'KAZ': set([1999]), u'BEN':
set([1994]), u'NGA': set([1997, 1989, 1983]), u'ZWE': set([2003, 1991, 1998, 1983]),
u'LKA': set([1978]), u'MWI': set([1994]), u'CRI': set([1981, 1991]), u'CMR': set([1994]),
u'COM': set([1994]), u'UGA': set([1988, 1980]), u'TKM': set([1993]), u'TTO': set([1986]),
u'TCD': set([1994]), u'GEO': set([1992, 1999]), u'ROU': set([1996]), u'MNG': set([1997,
1990]), u'BLR': set([1994, 1999]), u'GRC': set([1983]), u'MOZ': set([1987]), u'ARG':
set([2002, 1987, 1981, 1975]), u'TJK': set([1999]), u'HTI': set([1992, 2003]), 'STP':
set([1992, 1987, 1997]), u'VNM': set([1987, 1972, 1981]), u'FJI': set([1998]), u'HND':
set([1990]), u'DOM': set([1985, 2003, 1990]), u'ISR': set([1985, 1980, 1975]), u'PER':
set([1976, 1988, 1981]), u'IDN': set([1979, 1998]), u'SUR': set([2001, 1995, 1990]),
'COG': set([1976, 1994, 1999, 1989, 1983]), u'ISL': set([1989, 1981, 1975]), u'ETH':
set([1993]), u'NER': set([1994]), u'COL': set([1985]), u'BWA': set([1984]), u'MDA':
set([1999]), u'MDG': set([1984, 1994, 2004]), u'ECU': set([1982, 1999]), u'SEN':
set([1994]), u'MDV': set([1975]), 'SRB': set([2000]), u'LTU': set([1992]), u'RWA':
set([1991]), u'ZMB': set([1996, 1989, 1983]), u'GMB': set([1985, 2003]), u'GTM':
set([1986]), u'UKR': set([1998]), 'VEN': set([1984, 1994, 2002, 1989]), u'KEN':
set([1993]), 'LAO': set([1978, 1972, 1986, 1997]), u'TUR': set([1984, 2001, 1978, 1996,
1991]), u'ALB': set([1997]), u'MMR': set([2001, 1996, 1990, 1975]), 'RUS': set([1998]),
u'MEX': set([1977, 1995, 1982]), u'BRA': set([1976, 1992, 1987, 1982, 1999]), u'GNQ':
set([1994, 1980]), u'SWE': set([1993]), u'AZE': set([1994]), u'GNB': set([1994, 1980]),
u'SWZ': set([1985]), 'CIV': set([1994]), u'GUY': set([1987]), 'KOR': set([1998]), 'CAF':
set([1994]), u'UZB': set([2000, 1994]), u'GAB': set([1994]), u'EST': set([1992]), u'ESP':
set([1983]), u'SLV': set([1986]), u'MLI': set([1994]), u'LVA': set([1992]), 'IRN': set([2000,
1985, 1993]), u'SLE': set([1989, 1998, 1983]), u'ITA': set([1981]), u'KGZ': set([1997]),
u'LSO': set([1985]), u'SDN': set([1994, 1988, 1981]), u'ARM': set([1994]), u'PNG':
set([1995])}
```

In [15]:
```
print(currency_crises["DZA"])
```

```
set([1994, 1988])
```

Doing stuff to data

For starters here's my heuristic for generating normal years.

In [16]:
```
def pick_normal_years(crisis_years, minimum = 1971, maximum = 2007):
  crisis_years = sort(list(crisis_years))
  normal_years = []
  safe_from_crisis = 10
  if 0<len(crisis_years)<=3:
    # before
    year_before = crisis_years[0]-safe_from_crisis
    while year_before>=minimum:
      normal_years.append(year_before)
      year_before-=safe_from_crisis
```

```
    # in between
    for i in range(len(crisis_years)-1):
      if crisis_years[i+1]-crisis_years[i]>=20:
        delta = crisis_years[i+1]-crisis_years[i]
        normal_years.append(crisis_years[i]+int(round(delta/2.0)))
    # after
    year_after = crisis_years[-1]+safe_from_crisis
    while year_after <= maximum:
      normal_years.append(year_after)
      year_after+=safe_from_crisis
  return sort(normal_years)
print(pick_normal_years([1982, 1983, 2004]))
print(pick_normal_years([1983, 1993]))
print(pick_normal_years([2003, 2004]))
print(pick_normal_years([1973, 1974]))
```

```
[1972 1994]
[1973 2003]
[1973 1983 1993]
[1984 1994 2004]
```

A function to combine crises

In [17]:

```
def combine_crises(crisis_def_list):
  result_crisis_def = {}
  for crisis_def in crisis_def_list:
    for country in crisis_def.keys():
      years = crisis_def[country]
      try:
        result_crisis_def[country]|=years
      except KeyError:
        result_crisis_def[country]=years
  return result_crisis_def
crisis_def_a = {"HRV":set([1998, 2008]), "USA":set([1975, 2009])}
crisis_def_b = {"GBR":set([1979]), "USA":set([2011])}
print(combine_crises([crisis_def_a, crisis_def_b]))
```

```
{'HRV': set([2008, 1998]), 'GBR': set([1979]), 'USA': set([2009, 2011, 1975])}
```

Writing data

In [18]:

```
from xlwt import Workbook
def write_datatests(crisis_def_list, location = "./out", suffix = ""):
  book = Workbook()
  sheet1 = book.add_sheet('Sheet 1')
  result_crises = combine_crises(crisis_def_list)
  row_num = 0
  for country_code in sort(result_crises.keys()):
    years = sort(list(result_crises[country_code]))
    try:
      len(years)
    except:
```

```
            print(years)
            print(country_code)
        sheet1.write(row_num, 0, country_code)
        crisis_row = sheet1.row(row_num)
        crisis_row.write(1, "crisis")
        for j in range(len(years)):
            crisis_row.write(j+2, years[j])
        normal_row = sheet1.row(row_num+1)
        normal_row.write(1, "normal")
        normal_years = pick_normal_years(years)
        for j in range(len(normal_years)):
            normal_row.write(j+2, normal_years[j])
        row_num+=2
    saveloc = os.path.expanduser(location)+suffix+".xls"
    book.save(saveloc)
location="~/Dropbox/dev/itd/skripte/ipy_notebook/data/imf/crisis-imf-"
write_datatests([banking_crises], location, suffix = "banking")
write_datatests([currency_crises], location, suffix = "currency")
write_datatests([debt_crises], location, suffix = "debt")
write_datatests([banking_crises, currency_crises], location, suffix = "banking+currency
write_datatests([banking_crises, debt_crises], location, suffix = "banking+debt")
write_datatests([currency_crises, debt_crises], location, suffix = "currency+debt")
write_datatests([banking_crises, currency_crises, debt_crises], location, suffix = "banki
print("done")
```

```
done
```

Parse the new IMF dataset from 2012

Parse thew new data as dictionaries and write them down.

In [22]:
```
def csv_to_years(csv_years):
    if type(csv_years)==float or type(csv_years)==int:
        years = [int(csv_years)]
    else:
        years = list(int(x) for x in csv_years.split(", ") if x!="")
    return years

def add_many_crises(what, where, key):
    #where[key].add(what)
    if what!=[]:
        try:
            where[key] |= set(what) # union with the existing set
        except KeyError:
            where[key] = set(what)

def parse_imf_db_2012(loc):
    wb = xlrd.open_workbook(loc)
    sh = wb.sheet_by_index(1)
    banking_crises = {}
    currency_crises = {}
    debt_crises = {}
    problematic = []
```

```
  for rownum in range(2,sh.nrows):
    country = sh.row_values(rownum)[0].rstrip()
    try:
      code = country_codes[country]
    except KeyError:
      code = country # we'll fix those maunally
      problematic.append(code)
    csv_years_all = [x for x in sh.row_values(rownum)[1:4]]
    #years = [int(x) if x!="" else None for x in sh.row_values(rownum)[1:]]
    #print(csv_to_years(csv_years_all[0]))
    #print(sh.row_values(rownum))
    #print(rownum)
    add_many_crises(csv_to_years(csv_years_all[0]), banking_crises, code)
    add_many_crises(csv_to_years(csv_years_all[1]), currency_crises, code)
    add_many_crises(csv_to_years(csv_years_all[2]), debt_crises, code)
    #print(country, years)
  print("PROBLEMATIC:")
  print(problematic)
  return banking_crises, currency_crises, debt_crises
loc = os.path.expanduser("~/Dropbox/dev/itd/skripte/ipy_notebook/data/imf/2012/IMF
banking_crises_new, currency_crises_new, debt_crises_new = parse_imf_db_2012(loc)
```

```
PROBLEMATIC:
[u'Yugoslavia, SFR']
{u'DZA': set([1994, 1988]), u'AGO': set([1996, 1991]), 'EGY': set([1979, 1990]), u'BGD':
set([1976]), u'NAM': set([1984]), u'BGR': set([1996]), u'BOL': set([1973, 1981]), u'GHA':
set([2000, 1993, 1978, 2009, 1983]), u'PAK': set([1972]), u'JOR': set([1989]), u'LBY':
set([2002]), u'MYS': set([1998]), u'TZA': set([1985, 1990]), u'PRT': set([1983]), u'KHM':
set([1992, 1971]), u'PRY': set([1984, 2002, 1989]), u'LBN': set([1984, 1990]), u'BFA':
set([1994]), u'MRT': set([1993]), u'CHL': set([1972, 1982]), u'JAM': set([1978, 1991,
1983]), u'GIN': set([2005, 1982]), u'FIN': set([1993]), u'URY': set([2002, 1972, 1990,
1983]), u'THA': set([1998]), u'SYC': set([2008]), u'NPL': set([1984, 1992]), u'MAR':
set([1981]), 'YEM': set([1985, 1995]), u'PHL': set([1998, 1983]), u'ZAF': set([1984]),
u'NIC': set([1985, 1979, 1990]), u'TGO': set([1994]), u'SYR': set([1988]), u'KAZ':
set([1999]), u'BEN': set([1994]), u'NGA': set([1997, 1989, 1983]), u'ZWE': set([2003,
1991, 1998, 1983]), u'LKA': set([1978]), u'MWI': set([1994]), u'CRI': set([1981, 1991]),
u'CMR': set([1994]), u'COM': set([1994]), u'UGA': set([1988, 1980]), u'TKM': set([2008,
1993]), u'TTO': set([1986]), u'TCD': set([1994]), u'GEO': set([1992, 1999]), u'ROU':
set([1996]), u'MNG': set([1997, 1990]), u'BLR': set([2009, 1994, 1999]), u'GRC':
set([1983]), u'MOZ': set([1987]), u'ARG': set([2002, 1987, 1981, 1975]), u'TJK':
set([1999]), u'HTI': set([1992, 2003]), 'STP': set([1992, 1987, 1997]), u'VNM': set([1987,
1972, 1981]), u'FJI': set([1998]), u'HND': set([1990]), u'DOM': set([1985, 2003, 1990]),
u'ISR': set([1985, 1980, 1975]), u'PER': set([1976, 1988, 1981]), u'IDN': set([1979,
1998]), u'SUR': set([2001, 1995, 1990]), 'COG': set([1989, 1994, 1999, 1976, 2009,
1983]), u'ISL': set([2008, 1989, 1981, 1975]), u'ETH': set([1993]), u'NER': set([1994]),
u'COL': set([1985]), u'BWA': set([1984]), u'MDA': set([1999]), u'MDG': set([1984, 1994,
2004]), u'ECU': set([1982, 1999]), u'SEN': set([1994]), u'MDV': set([1975]), 'SRB':
set([2000]), u'LTU': set([1992]), u'RWA': set([1991]), u'ZMB': set([2009, 1996, 1989,
1983]), u'GMB': set([1985, 2003]), u'GTM': set([1986]), u'UKR': set([2009, 1998]), 'VEN':
set([1984, 2010, 1994, 2002, 1989]), u'KEN': set([1993]), 'LAO': set([1978, 1972, 1986,
1997]), u'TUR': set([1984, 2001, 1978, 1996, 1991]), u'ALB': set([1997]), u'MMR':
set([2001, 1996, 1990, 1975]), 'RUS': set([1998]), u'MEX': set([1977, 1995, 1982]),
u'BRA': set([1976, 1992, 1987, 1982, 1999]), u'GNQ': set([1994, 1980]), u'SWE':
set([1993]), u'AZE': set([1994]), u'GNB': set([1994, 1980]), u'SWZ': set([1985]), 'CIV':
set([1994]), u'GUY': set([1987]), 'KOR': set([1998]), 'CAF': set([1994]), u'UZB':
set([2000, 1994]), u'GAB': set([1994]), u'EST': set([1992]), u'ESP': set([1983]), u'SLV':
```

```
set([1986]), u'MLI': set([1994]), u'LVA': set([1992]), 'IRN': set([2000, 1985, 1993]),
u'SLE': set([1989, 1998, 1983]), u'NZL': set([1984]), u'ITA': set([1981]), u'KGZ':
set([1997]), u'NCL': set([1981]), u'LSO': set([1985]), u'SDN': set([1994, 1988, 1981]),
u'ARM': set([1994]), u'PNG': set([1995])}
```

In [20]:
```
location = os.path.expanduser("~/Dropbox/dev/itd/skripte/ipy_notebook/data/imf/2012
write_datatests([banking_crises_new], location, suffix = "banking")
write_datatests([currency_crises_new], location, suffix = "currency")
write_datatests([debt_crises_new], location, suffix = "debt")
write_datatests([banking_crises_new, currency_crises_new], location, suffix = "banking
write_datatests([banking_crises_new, debt_crises_new], location, suffix = "banking+de
write_datatests([currency_crises_new, debt_crises_new], location, suffix = "currency+
write_datatests([banking_crises_new, currency_crises_new, debt_crises_new], locatior
```

Find deltas...

In [26]:
```
def find_delta(crisis_def, crisis_def_new):
    crisis_def_delta = {}
    for country in crisis_def_new:
        years_new = crisis_def_new[country]
        try:
            years_old = crisis_def[country]
        except KeyError:
            years_old = set()
        added_years = years_new - years_old
        if len(added_years)>0:
            crisis_def_delta[country] =  added_years
    return crisis_def_delta
crisis_def_a = {"HRV":set([1998, 2008]), "USA":set([1975, 2009])}
crisis_def_b = {"GBR":set([1979]), "USA":set([1975, 2009,2011]), "HRV":set([1998, 2008
print(find_delta(crisis_def_a, crisis_def_b))
```

```
{'GBR': set([1979]), 'USA': set([2011])}
```

In [29]:
```
banking_crises_delta = find_delta(banking_crises, banking_crises_new)
currency_crises_delta = find_delta(currency_crises, currency_crises_new)
debt_crises_delta = find_delta(debt_crises, debt_crises_new)
print(banking_crises_delta)
write_datatests([banking_crises_delta], location, suffix = "delta-banking")
write_datatests([currency_crises_delta], location, suffix = "delta-currency")
write_datatests([debt_crises_delta], location, suffix = "delta-debt")
write_datatests([banking_crises_delta, currency_crises_delta], location, suffix = "delta-
write_datatests([banking_crises_delta, debt_crises_delta], location, suffix = "delta-bank
write_datatests([currency_crises_delta, debt_crises_delta], location, suffix = "delta-cur
write_datatests([banking_crises_delta, currency_crises_delta, debt_crises_delta], locat
```

```
{u'NGA': set([2009]), u'BEL': set([2008]), u'LUX': set([2008]), u'SWE': set([2008]),
u'DEU': set([2008]), u'ISL': set([2008]), u'HUN': set([2008]), u'PRT': set([2008]), u'NLD':
set([2008]), u'MNG': set([2008]), u'SVN': set([2008]), u'FRA': set([2008]), u'CHE':
set([2008]), u'ESP': set([2008]), u'DNK': set([2008]), u'UKR': set([2008]), u'LVA':
set([2008]), u'IRL': set([2008]), u'AUT': set([2008]), 'RUS': set([2008]), u'ITA':
set([2008]), u'KAZ': set([2008]), u'GRC': set([2008])}
```