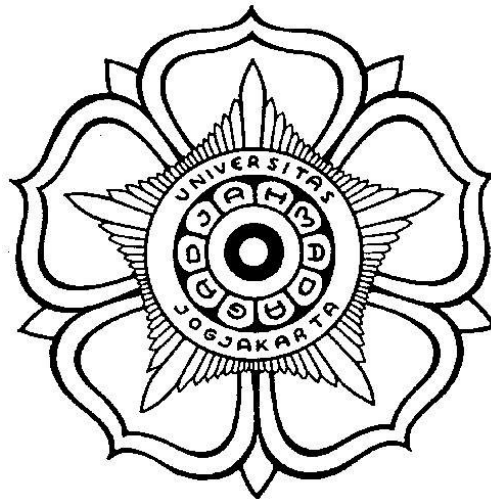


**LAMPIRAN SKRIPSI**

**PENGEMBANGAN INPUT METHOD EDITOR  
BAHASA JEPANG BERBASIS WEB**

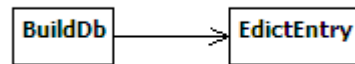
**Agro Rachmatullah  
03/165399/PA/09374**



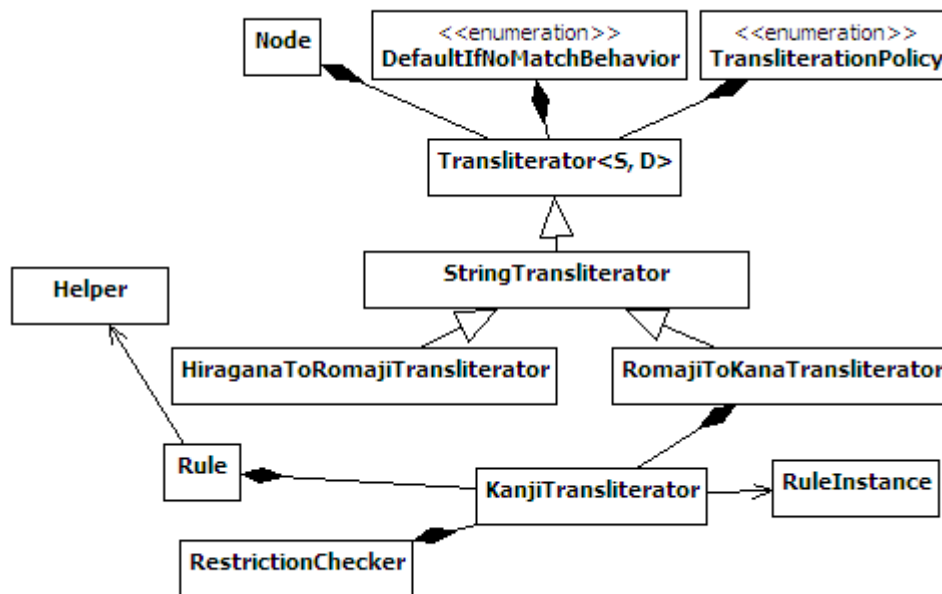
**DEPARTEMEN PENDIDIKAN NASIONAL  
UNIVERSITAS GADJAH MADA  
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM  
YOGYAKARTA  
2007**



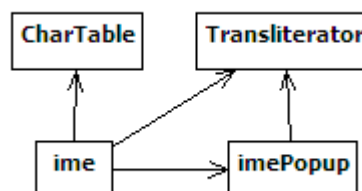
## LAMPIRAN A - Diagram Kelas



Gambar A.1 Diagram Kelas-Kelas pada EdictDbBuilder.exe



Gambar A.2 Diagram Kelas-Kelas pada GamaIme.dll



Gambar A.3 Diagram Kelas-Kelas pada Aplikasi Web IME

## LAMPIRAN B - Source Code

### B.1 EdictDbBuilder.exe

#### B.1.1 Edict.cs

```

using System.Text;
using System.Collections.Generic;
using System.IO;
using System;
using System.Reflection;

namespace EdictDbBuilder
{
    public class EdictEntry
    {
        string kana;
        string kanji;
        string[] tags;
        int popularity = 0;
        bool isUsuallyKana = false;

        public static string[] AllowedTags;
        public static Dictionary<string, int> PopularityTagDict;
        public static Dictionary<string, string>
TagTransformationDict;

        static EdictEntry()
        {
            List<string> allowedTagList = new List<string>();
            using (Stream fs =
Assembly.GetAssembly(typeof(EdictEntry)).GetManifestResourceStrea
m("EdictDbBuilder.edict_allowed_tags.txt"))
            {
                using (StreamReader sr = new StreamReader(fs))
                {
                    string line;
                    while((line = sr.ReadLine()) != null)
                    {
                        if(!(line.Length == 0) && !line.StartsWith("%"))
                        {
                            allowedTagList.Add(line);
                        }
                    }
                }
            }
            allowedTagList.Sort();
            AllowedTags = allowedTagList.ToArray();

            PopularityTagDict = new Dictionary<string, int>();
            using (Stream fs =
Assembly.GetAssembly(typeof(EdictEntry)).GetManifestResourceStrea

```

```

m("EdictDbBuilder.edict_popularity_tags.txt"))
    {
        using (StreamReader sr = new StreamReader(fs))
        {
            string line;
            while((line = sr.ReadLine()) != null)
            {
                if(!(line.Length == 0) && !line.StartsWith("%"))
                {
                    string[] tokens = line.Split('\t');
                    PopularityTagDict[tokens[0]] =
int.Parse(tokens[1]);
                }
            }
        }

        TagTransformationDict = new Dictionary<string, string>();
        using (Stream fs =
Assembly.GetAssembly(typeof(EdictEntry)).GetManifestResourceStream("EdictDbBuilder.edict_tags_transform.txt"))
        {
            using (StreamReader sr = new StreamReader(fs))
            {
                string line;
                while((line = sr.ReadLine()) != null)
                {
                    if(!(line.Length == 0) && !line.StartsWith("%"))
                    {
                        string[] tokens = line.Split('\t');
                        TagTransformationDict[tokens[0]] = tokens[1];
                    }
                }
            }
        }
    }
public string Kanji
{
    get
    {
        return this.kanji;
    }
}
public string Kana
{
    get
    {
        return this.kana;
    }
}
public string[] Tags
{
    get
    {
        return this.tags;
    }
}

```

```

    }
    public int Popularity
    {
        get
        {
            return this.popularity;
        }
    }
    public bool IsUsuallyKana
    {
        get
        {
            return this.isUsuallyKana;
        }
    }
    public EdictEntry(string edictLine)
    {
        int slashIndex = edictLine.IndexOf('/');
        string translation = edictLine.Substring(slashIndex);
        int bracketIndex = edictLine.IndexOf('[');
        if (bracketIndex == -1)
        {
            this.kanji = edictLine.Substring(0, slashIndex - 1);
            this.kana = KatakanaToHiragana(this.kanji);
        }
        else
        {
            this.kanji = edictLine.Substring(0, bracketIndex - 1);
            this.kana =
KatakanaToHiragana(edictLine.Substring(bracketIndex + 1,
slashIndex - bracketIndex - 3));
        }

        // read tags
        List<string> tagList = new List<string>();
        int startIndex = -1;
        bool inParanthesis = false;
        for(int i = 0; i < translation.Length; i++)
        {
            char c = translation[i];
            if(c == '(')
            {
                inParanthesis = true;
                startIndex = i + 1;
            }
            else if(c == ',' && inParanthesis)
            {
                string tag = translation.Substring(startIndex, i -
startIndex);
                this.TryAddTag(tagList, tag);
                startIndex = i + 1;
            }
            else if(c == ')')
            {
                string tag = translation.Substring(startIndex, i -
startIndex);

```

```

        this.TryAddTag(tagList, tag);
        inParanthesis = false;
    }
}
this.tags = tagList.ToArray();
}
private void TryAddTag(List<string> tagList, string tag)
{
    if(Array.BinarySearch(AllowedTags, tag) >= 0)
    {
        tagList.Add(tag);
    }
    else if(tag == "uk")
    {
        this.isUsuallyKana = true;
    }
    else if(PopularityTagDict.ContainsKey(tag))
    {
        this.popularity += PopularityTagDict[tag];
    }
    else if(TagTransformationDict.ContainsKey(tag))
    {
        TryAddTag(tagList, TagTransformationDict[tag]);
    }
}

public static string KatakanaToHiragana(string input)
{
    StringBuilder sb = new StringBuilder(input.Length);
    foreach(char c in input)
    {
        if(0x30a1 <= c && c <= 0x30f4)
        {
            sb.Append((char)(c - 0x60));
        }
        else
        {
            sb.Append(c);
        }
    }
    return sb.ToString();
}
}
}

```

### B.1.2 BuildDb.cs

```

using System.Collections.Generic;
using System.Data.SQLite;
using System.Text;
using System.IO;
using System;

```

```
namespace EdictDbBuilder
{
    class BuildDb
    {
        static string edictFile = "edict";
        static string outputDbFile = "edict.db3";

        static int Main(string[] args)
        {
            int nextArg = 0;
            if(args.Length > nextArg)
            {
                if(args[nextArg] == "/" || args[nextArg] == "-h"
                    || args[nextArg] == "--help")
                {
                    Help();
                    return 0;
                }
                if(args[nextArg] == "--")
                {
                    nextArg++;
                }
                edictFile = args[nextArg++];
            }

            if(args.Length > nextArg)
            {
                outputDbFile = args[nextArg];
            }

            List<EdictEntry> entries = new List<EdictEntry>();

            try
            {
                using(StreamReader sr = new StreamReader(new
                    FileStream(edictFile, FileMode.Open, FileAccess.Read),
                    Encoding.GetEncoding("euc-jp")))
                {
                    string input;
                    sr.ReadLine(); // the first line is edict version

                    while ((input = sr.ReadLine()) != null)
                    {
                        EdictEntry entry = new EdictEntry(input);
                        entries.Add(entry);
                    }
                }
            }
            catch(Exception e)
            {
                Console.Error.WriteLine("Error reading input file:");
                Console.Error.WriteLine(e.Message);
                return 1;
            }
        }
    }
}
```

```

        if (File.Exists (outputDbFile))
        {
            try
            {
                File.Delete (outputDbFile);
            }
            catch (Exception e)
            {
                Console.Error.WriteLine ("Error creating output
file:");
                Console.Error.WriteLine (e.Message);
                return 1;
            }
        }

        SQLiteConnection conn = new SQLiteConnection ("Data
Source=" + outputDbFile);
        PopulateDb (conn, entries);

        return 0;
    }

    static void Help ()
    {
        Console.WriteLine ("This programs reads a valid EDICT file
and");
        Console.WriteLine ("outputs an SQLite file for use by Gama
IME.");
        Console.WriteLine ();
        Console.WriteLine ("Syntax:");
        Console.WriteLine ("EdictDbBuilder.exe [edictFile]
[outputFile]");
        Console.WriteLine ("Default values:");
        Console.WriteLine ("[edictFile]: \"{0}\"", edictFile);
        Console.WriteLine ("[outputFile]: \"{0}\"", outputDbFile);
    }

    static void PopulateDb (SQLiteConnection conn,
List<EdictEntry> entries)
    {
        try
        {
            StringBuilder errorReport = new StringBuilder ();

            conn.Open ();

            SQLiteCommand cmd = new SQLiteCommand (conn);
            cmd.CommandText = "CREATE TABLE edict (kana STRING,
kanji STRING, tags STRING, pop INTEGER, uk BOOL, PRIMARY KEY
(kana, kanji));";
            cmd.ExecuteNonQuery ();

            cmd.CommandText = "BEGIN;";
            cmd.ExecuteNonQuery ();

            int counter = 0;

```

```

        int errorCounter = 0;

        foreach(EdictEntry entry in entries)
        {
            cmd.CommandText = String.Format("INSERT INTO
edict values('{0}', '{1}', '{2}', {3}, {4});",
                                                    entry.Kana,
                                                    entry.Kanji,
JoinTags(entry.Tags),
                                                    entry.Popularity,
entry.IsUsuallyKana ? '1' : '0');
            try
            {
                // primary key constraint can be violated
because of
                // 1) Separate entries in EDICT that should be
one, e.g.:
                // めん棒 [めんぼう] /(n) cotton swab/
                // めん棒 [めんぼう] /(n) rolling pin/

                // 2) Alternate kana spelling in EDICT, which
                //   in this database generation is all converted
                //   to hiragana, e.g.:
                // 桃 [もも] /(n) peach/prunus persica (tree)/(P)/
                // 桃 [モモ] /(n) peach/prunus persica (tree)/

                // in case of violation, only the first one will be
added.

                cmd.ExecuteNonQuery();
                counter++;
                if(counter % 500 == 0)
                {
                    Console.WriteLine("Adding {0}:{1} (entry
{2})", entry.Kana, entry.Kanji, counter);
                }
            }
            catch(Exception e)
            {
                errorCounter++;
                errorReport.AppendLine("=====");
                errorReport.AppendLine(e.Message);

                errorReport.AppendLine(String.Format("Problematic entry: {0}:
{1}", entry.Kana, entry.Kanji));
            }
        }

        cmd.CommandText = "COMMIT;";
        cmd.ExecuteNonQuery();

        Console.WriteLine("Done Adding.");

```

```

        Console.WriteLine();

        if(errorCounter > 0)
        {
            Console.Error.WriteLine("The following {0} error{1}
encountered:", errorCounter, errorCounter == 1 ? " was" : "s
were");
            Console.Error.WriteLine(errorReport);
        }

        Console.WriteLine("{0} items added.", counter);

    }
    finally
    {
        if(conn != null)
        {
            conn.Close();
        }
    }
}

static string JoinTags(string[] tags)
{
    StringBuilder sb = new StringBuilder("|");
    foreach(string input in tags)
    {
        sb.Append(input).Append("|");
    }
    return sb.ToString();
}
}
}

```

### B.1.3 edict\_allowed\_text.txt

```

adj
adv
adj-na
adj-no
adj-pn
adj-t
aux
aux-v
conj
exp
int
n
n-adv
n-t
n-suf
n-pref
num

```

```

pref
prt
suf
v1
v5u
v5u-s
v5k
v5g
v5s
v5t
v5n
v5b
v5m
v5r
v5k-s
v5aru
v5uru
vi
vs
vs-i
vs-s
vz
vk

%not existing in documentation
adv-to

```

#### B.1.4 edict\_tags\_transform.txt

```
v5r-i v5r
```

#### B.1.5 edict\_popularity\_tags.txt

```

% Every EDICT item is considered to have popularity 0 (neutral)
% Tags defined in this file will modify the popularity
% There can be many of these tags per item, so it is additive

% The tags themselves will "dissapear",
% i.e., is not represented in the item's tag list

% Format:
% [tag][tab][value]

P 2
arch -1
iK-1
ik-1
io-1
obs -1
obsc -1
oK-1
ok-1

```

#### B.2 GamaIme.dll

### B.2.1 Helper.cs

```

using System;
using System.Text;
using System.Collections;

public class Helper
{
    public static string JoinArray(IEnumerable array, string
separator)
    {
        StringBuilder sb = new StringBuilder();
        foreach(object a in array)
        {
            sb.Append(a).Append(separator);
        }
        if(sb.Length > 0)
        {
            sb.Remove(sb.Length - 1, 1);
        }
        return sb.ToString();
    }
}

```

### B.2.2 HiraganaToRomajiTransliterator.cs

```

using System.IO;
using System.Reflection;

namespace GamaIme
{
    /// <summary>
    /// Provides reversible hiragana to romaji transliteration
sevice.
    /// </summary>
    /// <remarks>
    /// Choices are deliberately made to ease grammatical rule
construction,
    /// so for example つ is transliterated as tu.
    /// </remarks>
    public class HiraganaToRomajiTransliterator :
StringTransliterator
    {
        public HiraganaToRomajiTransliterator()
        {
            using (Stream fs =
Assembly.GetAssembly(typeof(HiraganaToRomajiTransliterator)).GetM
anifestResourceStream("GamaIme.hiragana_to_romaji.txt"))
            {
                using (StreamReader sr = new StreamReader(fs))
                {
                    this.ParseRule(sr.ReadToEnd());
                }
            }
        }
    }
}

```

```

    }
}
}
}

```

### B.2.3 KanjiTransliterator.cs

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Text;
using System.Data.SQLite;

namespace GamaIme
{
    public class KanjiTransliterator
    {
        public static RomajiToKanaTransliterator kanaTransliterator =
new RomajiToKanaTransliterator();
        RestrictionChecker restrictionChecker;
        Rule[] rules;
        string[] edictTags;
        string dbFileName;
        public KanjiTransliterator(string ruleFileName, string
restrictionFileName, string tagFileName, string dbFileName)
        {
            this.restrictionChecker =
RestrictionChecker.Load(restrictionFileName);
            this.rules = Rule.Load(ruleFileName);
            this.edictTags = File.ReadAllLines(tagFileName);
            this.dbFileName = dbFileName;
        }
        public string[] Transliterate(string inputRomaji)
        {
            // only RuleInstance objects with edict_tags srcTag will
be added here
            // RuleInstance objects here will be used to search the
database
            List<List<RuleInstance>> instancesByLevel = new
List<List<RuleInstance>>();
            Queue<RuleInstance> queue = new Queue<RuleInstance>();

            RuleInstance perfectMatch = new RuleInstance(inputRomaji);

            instancesByLevel.Add(new List<RuleInstance>(1));
            instancesByLevel[0].Add(perfectMatch);
            queue.Enqueue(perfectMatch);

            Console.WriteLine(perfectMatch);

            while(queue.Count != 0)
            {
                instancesByLevel.Add(new List<RuleInstance>());
            }
        }
    }
}

```

```

        Console.WriteLine("-- searching next level of tree...
--");
        int currentLevelAmount = queue.Count;
        for(int i = 0; i < currentLevelAmount; i++)
        {
            RuleInstance instance = queue.Dequeue();
            List<RuleInstance> nextInstances =
this.GetNextInstances(instance);
            foreach(RuleInstance nextInstance in nextInstances)
            {
                Console.WriteLine(nextInstance);
                queue.Enqueue(nextInstance);
                if(Array.IndexOf(this.edictTags,
                    nextInstance.srcTag) != -1)
                {
                    instancesByLevel[instancesByLevel.Count -
1].Add(nextInstance);
                }
                else
                {
                    Console.WriteLine(nextInstance.srcWordKana + "
WILL NOT BE USED FOR DB QUERY (because tag is " +
nextInstance.srcTag + ")");
                }
            }
        }
        if(instancesByLevel[instancesByLevel.Count - 1].Count ==
0)
        {
            instancesByLevel.RemoveAt(instancesByLevel.Count - 1);
        }

        DateTime time = DateTime.Now;
        Console.WriteLine("-- FetchDb start");
        string[] result = this.FetchDb(instancesByLevel);
        Console.WriteLine("-- FetchDb search time: " +
(DateTime.Now - time));
        return result;
    }

    public string[] FetchDb(List<List<RuleInstance>>
instancesByLevel)
    {
        // get db rows
        List<string> tagsToSearch = new List<string>();

        StringBuilder query = new StringBuilder("SELECT * FROM
edict WHERE ");

        // search from the deepest
        // because we assume a short one (e.g. い<)
        // is more likely to be found than a longer one (e.g. いか
な<い)

```

```

for(int i = instancesByLevel.Count - 1; i >= 0; i--)
{
    foreach(RuleInstance instance in instancesByLevel[i])
    {
        string srcWordKana = instance.srcWordKana;
        if(tagsToSearch.IndexOf(srcWordKana) >= 0)
        {
            continue;
        }
        tagsToSearch.Add(srcWordKana);
        query.Append("kana='");
        query.Append(srcWordKana);
        query.Append("' OR ");
    }
}

query.Length -= 4;

Console.WriteLine(query);

SQLiteConnection conn = null;
SQLiteDataReader reader = null;

List<string> result = new List<string>();
Dictionary<int, List<string>>[] popResultByLevel = new
Dictionary<int, List<string>>[instancesByLevel.Count];
for(int i = 0; i < popResultByLevel.Length; i++)
{
    popResultByLevel[i] = new Dictionary<int,
List<string>>();
}

bool uk = false; // marker whether uk is already added, if
any
try
{
    conn = new SQLiteConnection("Data Source=" +
this.dbFileName);
    conn.Open();
    SQLiteCommand cmd = new SQLiteCommand(query.ToString(),
conn);
    reader = cmd.ExecuteReader();
    while(reader.Read())
    {
        for(int i = 0; i < instancesByLevel.Count; i++)
        {
            foreach(RuleInstance instance in
instancesByLevel[i])
            {
                if((string)reader["kana"] ==
instance.srcWordKana)
                {
                    if(instance.srcTag == "any" ||
((string)reader["tags"]).Contains("|" + instance.srcTag + "|"))
                    {

```



```

        if(toPut > keys[toPutIndex])
        {
            int temp = keys[toPutIndex];
            keys[toPutIndex] = toPut;
            toPut = temp;
        }
        toPutIndex++;
    }
    keys[nextIndex] = toPut;
    nextIndex++;
}

foreach(int k in keys)
{
    foreach(string s in popResult[k])
    {
        if(!result.Contains(s))
        {
            result.Add(s);
        }
    }
}

return result.ToArray();
}

public static string HiraganaToKatakana(string input)
{
    StringBuilder sb = new StringBuilder(input.Length);
    foreach(char c in input)
    {
        if(0x30a1 - 0x60 <= c && c <= 0x30f4 - 0x60)
        {
            sb.Append((char)(c + 0x60));
        }
        else
        {
            sb.Append(c);
        }
    }
    return sb.ToString();
}

static void AddDbResult(Dictionary<int, List<string>>
popResult, string trans, int pop)
{
    if(popResult.ContainsKey(pop))
    {
        popResult[pop].Add(trans);
    }
    else
    {
        popResult[pop] = new List<string>();
        popResult[pop].Add(trans);
    }
}

```

```

    }
}

public List<RuleInstance> GetNextInstances(RuleInstance
instance)
{
    List<RuleInstance> instances = new List<RuleInstance>();
    foreach(Rule rule in this.rules)
    {
        if(instance.srcTag != "any" &&
            Array.IndexOf(rule.destTags, instance.srcTag) == -1)
        {
            continue;
        }

        // to prevent empty string from fulfilling rules such as
        // * -> *da
        if(rule.srcPatternEnd.Length == 0 &&
            (rule.destPatternStartPlain.Length +
            rule.destPatternEndPlain.Length + 1
            >
            instance.srcWordRomaji.Length))
        {
            continue;
        }

        if(!
            instance.srcWordRomaji.StartsWith(rule.destPatternStartPlain))
        {
            continue;
        }

        if(!
            instance.srcWordRomaji.EndsWith(rule.destPatternEndPlain))
        {
            continue;
        }

        // rule applies!!!
        RuleInstance nextInstance = new RuleInstance();
        nextInstance.srcTag = rule.srcTag;
        nextInstance.srcWordRomaji =

            instance.srcWordRomaji.Substring(rule.destPatternStartPlain.Length,
            instance.srcWordRomaji.Length -
            rule.destPatternEndPlain.Length -
            rule.destPatternStartPlain.Length) +
            rule.srcPatternEndPlain;

        if(!this.restrictionChecker.Check(nextInstance.srcTag,

```

```

nextInstance.srcWordRomaji))
    {
        continue;
    }

    nextInstance.srcWordKana =
kanaTransliterator.Transliterate(nextInstance.srcWordRomaji);

    // the hard part: rule chaining

    // figure out the start part (e.g., go-houbi)

    string insertedStartNewKana =
kanaTransliterator.Transliterate(rule.destPatternStartPlain);
    nextInstance.startNewKana = instance.startNewKana +
insertedStartNewKana;
    if(instance.startNewKanji != null)
    {
        nextInstance.startNewKanji = instance.startNewKanji +
kanaTransliterator.Transliterate(GetKanjiPattern(rule.destPattern
Start));
    }
    else if(rule.destPatternStart.Length !=
rule.destPatternStartPlain.Length)
    {
        nextInstance.startNewKanji = instance.startNewKana +
kanaTransliterator.Transliterate(GetKanjiPattern(rule.destPattern
Start));
    }

    // figure out the ending part (the hard part!!!)

    // from the kana, analyze where the discrepancy
begins...

    // previously, for input ださなきゃ, it will be changed
into
    // ださなきゃ (original) - elementary RuleInstance
    // ださなければ (ければ/きゃ) - derived RuleInstance 1
    // ださなける (る/CRASH) - trying to derive RuleInstance 2
    // therefore the variable one line below is introduced
    int oldInstDiffIndex = instance.srcWordKana.Length -
instance.endOrigKana.Length;

    int differStart = 0; // this index iterates over
nextInstance.srcWordKana
    int instDiffIndex = insertedStartNewKana.Length; // this
index iterates over instance.srcWordKana
    for(;differStart < nextInstance.srcWordKana.Length &&
instDiffIndex < oldInstDiffIndex &&
nextInstance.srcWordKana[differStart] ==
instance.srcWordKana[instDiffIndex];
differStart++, instDiffIndex++)
    {
    }

```

```

// now find sound-morphing rule, such as (su)ru ->
(si)ta

bool soundMorph = rule.srcPatternEnd.StartsWith("(");
nextInstance.soundMorph = soundMorph;

// what needs to be cut from the EDICT entry
// example: する, る (for たべる -> たべた)
nextInstance.endOrigKana =
nextInstance.srcWordKana.Substring(differStart);

// what needs to be added based on the rule
// example: しない, た (for たべる -> たべた)
nextInstance.endNewKana =
instance.srcWordKana.Substring(instDiffIndex);

// rule chaining
nextInstance.endNewKana =
nextInstance.endNewKana.Substring(0,
nextInstance.endNewKana.Length - instance.endOrigKana.Length);
nextInstance.endNewKana += instance.endNewKana;

// the kanji replacement

bool kanjiReplacement = !soundMorph &&
rule.destPatternEnd.StartsWith("(");

if(kanjiReplacement)
{
    nextInstance.endNewKanji =
kanaTransliterater.Transliterate(GetKanjiPattern(rule.destPattern
End));

// rule chaining for kanji replacement
int soundMorphModifier = 0;
// to anticipate when the PREVIOUS instance has sound
morph
// e.g., もってくる (くる/きた) (sound morph in 来)
// (nextInstance's root word is もって)
if(instance.soundMorph)
{
    soundMorphModifier = 1;
}
nextInstance.endNewKanji =
nextInstance.endNewKanji.Substring(0,
nextInstance.endNewKanji.Length - instance.endOrigKana.Length +
soundMorphModifier);
    string addition = instance.endNewKanji == null ?
instance.endNewKana : instance.endNewKanji;
    nextInstance.endNewKanji +=
addition.Substring(soundMorphModifier);
}
else if(instance.endNewKanji != null)
{
    nextInstance.endNewKanji =
instance.srcWordKana.Substring(instDiffIndex);

```

```

        // rule chaining
        nextInstance.endNewKanji =
nextInstance.endNewKanji.Substring(0,
nextInstance.endNewKanji.Length - instance.endOrigKana.Length);
        nextInstance.endNewKanji += instance.endNewKanji;
    }

    instances.Add(nextInstance);
}

return instances;
}

static string GetKanjiPattern(string startPattern)
{
    StringBuilder sb = new StringBuilder();
    StringBuilder tempKana = null;
    bool copyChar = true;
    for(int i = 0; i < startPattern.Length; i++)
    {
        if(startPattern[i] == '(')
        {
            tempKana = new StringBuilder();
            copyChar = false;
            continue;
        }
        if(startPattern[i] == '|')
        {
            copyChar = true;
            continue;
        }
        if(startPattern[i] == ')')
        {
            if(!copyChar) // this is a sound morph rule, no kanji
replacement
            {
                copyChar = true;
                sb.Append(tempKana);
            }
            continue;
        }
        if(copyChar)
        {
            sb.Append(startPattern[i]);
        }
        else
        {
            tempKana.Append(startPattern[i]);
        }
    }
    return sb.ToString();
}

} // end of class

```

```
} // end of namespace
```

### B.2.4 RestrictionChecker.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Text.RegularExpressions;

namespace GamaIme
{
    public class RestrictionChecker
    {
        Dictionary<string, string[]> restriction;
        public RestrictionChecker(Dictionary<string, string[]>
restriction)
        {
            this.restriction = restriction;
        }
        public static RestrictionChecker Load(string fileName)
        {
            Dictionary<string, string[]> restriction = new
Dictionary<string, string[]>();

            string[] restrictionFileLines =
File.ReadAllLines(fileName);
            foreach(string line in restrictionFileLines)
            {
                Regex restrictionFormat = new Regex(@"^(?<tags>[a-zA-
Z0-9-]+(,[a-zA-Z0-9-]+)*):(?<endings>[a-z']+(,[a-z']+)*)$");
                if(line.Length != 0 && !line.StartsWith("%"))
                {
                    Match match = restrictionFormat.Match(line);
                    if(!match.Success)
                    {
                        throw new ArgumentException("The restriction \"" +
line + "\" on file \"" + fileName + "\" has an invalid format.",
"fileName");
                    }

                    string[] endings =
match.Groups["endings"].Value.Split(',');
                    for(int i = 0; i < endings.Length; i++)
                    {
                        if(Array.LastIndexOf(endings, endings[i]) != i)
                        {
                            throw new ArgumentException("The ending \"" +
endings[i] + "\" on restriction \"" + line + "\" on file \"" +
fileName + "\" appears more than once.", "fileName");
                        }
                    }

                    string[] tags = match.Groups["tags"].Value.Split(',');
                    foreach(string tag in tags)
```

```

        {
            if(restriction.ContainsKey(tag))
            {
                throw new ArgumentException("The tag \"" + tag +
"\\" on restriction \"" + line + "\" on file \"" + fileName + "\"
appears more than once.", "fileName");
            }
            restriction[tag] = endings;
        }
    }
}

return new RestrictionChecker(restriction);
}

public bool Check(string type, string word)
{
    if(this.restriction.ContainsKey(type))
    {
        foreach(string ending in this.restriction[type])
        {
            if(word.EndsWith(ending))
            {
                return true;
            }
        }
        // does not comply with any of the restriction

        return false;
    }
    else // no restriction placed
    {
        return true;
    }
}
}
}
}

```

### B.2.5 RomajiToKanaTransliterato.cs

```

using System.IO;
using System.Reflection;

namespace GamaIme
{
    /// <summary>
    /// Provides romaji to kana transliteration service.
    /// </summary>
    /// <remarks>A sequence of characters that cannot be
transliterated will remain as is.</remarks>
    /// <example>
    /// <code>
    /// RomajiToKanaTransliterato transliterato = new
RomajiToKanaTransliterato();

```

```

    /// transliterator.Transliterate("watashi"); // returns わたし
    /// transliterator.Transliterate("RABU"); // returns ラブ
    /// transliterator.Transliterate("xyz"); // returns xyz
    /// </code>
    /// </example>
    public class RomajiToKanaTransliterator :
StringTransliterator
    {
        public RomajiToKanaTransliterator()
        {
            using (Stream fs =
Assembly.GetAssembly(typeof(RomajiToKanaTransliterator)).GetManif
estResourceStream("GamaIme.romaji_to_hiragana.txt"))
            {
                using (StreamReader sr = new StreamReader(fs))
                {
                    this.ParseRule(sr.ReadToEnd());
                }
            }
            using (Stream fs =
Assembly.GetAssembly(typeof(RomajiToKanaTransliterator)).GetManif
estResourceStream("GamaIme.romaji_to_katakana.txt"))
            {
                using (StreamReader sr = new StreamReader(fs))
                {
                    this.ParseRule(sr.ReadToEnd());
                }
            }
        }
    }
}

```

### B.2.6 Rule.cs

```

using System;
using System.Text;
using System.Text.RegularExpressions;
using System.IO;
using System.Collections.Generic;

namespace GamaIme
{
    public class Rule
    {
        public string srcTag;
        public string srcPatternEnd;
        public string[] destTags;
        public string destPatternStart;
        public string destPatternEnd;

        public string srcPatternEndPlain;
        public string destPatternStartPlain;
        public string destPatternEndPlain;
    }
}

```



```

+)?)\))*)))/(?<destTags>[a-z0-9-]+(,[a-z0-9-]+)*)$");

    Match match = regex.Match(ruleString);

    if(!match.Success)
    {
        throw new ArgumentException("The rule \"" + ruleString +
"\\" is not in the correct format.", "ruleString");
    }

    string[] srcTags =
match.Groups["srcTags"].Value.Split(',');
    string[] newDestTags =
match.Groups["destTags"].Value.Split(',');
    string srcPatternEnd =
match.Groups["srcPatternEnd"].Value;
    string destPatternStart =
match.Groups["destPatternStart"].Value;
    string destPatternEnd =
match.Groups["destPatternEnd"].Value;

    Rule[] rules = new Rule[srcTags.Length];

    for(int i = 0; i < srcTags.Length; i++)
    {
        string srcTag = srcTags[i];
        if(Array.LastIndexOf(srcTags, srcTag) != i)
        {
            throw new ArgumentException("The source tag \"" +
srcTag + "\" on rule \"" + ruleString + "\" appears more than
once.", "ruleString");
        }
        string[] destTags = new string[newDestTags.Length + 1];
        destTags[0] = srcTag;
        for(int j = 0; j < newDestTags.Length; j++)
        {
            if(Array.IndexOf(destTags, newDestTags[j]) != -1)
            {
                throw new ArgumentException("The new destination
tag \"" + newDestTags[j] + "\" on rule \"" + ruleString + "\" is
the same as the source tag or appears more than once.",
"ruleString");
            }
            destTags[j + 1] = newDestTags[j];
        }

        rules[i] = new Rule(srcTag, srcPatternEnd,
destPatternStart,
                                destPatternEnd, destTags);

    }

    return rules;
}

public static Rule[] Load(string ruleFile)

```

```

    {
        List<Rule> rules = new List<Rule>();
        string[] fileLines = File.ReadAllLines(ruleFile);
        foreach(string line in fileLines)
        {
            // TODO: dummy debug code
            if(line == "TERMINATE")
            {
                break;
            }
            if(line.Length != 0 && !line.StartsWith("%"))
            {
                rules.AddRange(Rule.Parse(line));
            }
        }
        return rules.ToArray();
    }
}
}

```

### B.2.7 RuleInstance.cs

```

using System.Collections.Generic;
using System.Text;

namespace GamaIme
{
    public class RuleInstance
    {
        public string srcWordRomaji; // example: kau
        public string srcWordKana; // example: かう

        public string srcTag; // example: v5u (any to match any tag)

        public string startNewKana; // example: お
        public string startNewKanji; // example: 御, can be null if
there is no kanji alternative

        public string endOrigKana; // example: う
        public string endNewKana; // example: い
        public string endNewKanji; // example: null

        public bool soundMorph = false; // preserve a certain kanji
such as 来る -> 来た

        public override string ToString()
        {
            return this.srcTag + ": " + this.srcWordRomaji + " (" +
this.srcWordKana + ")[" + this.startNewKana + "/" +
this.startNewKanji + "]" + this.endOrigKana + "/" +
this.endNewKana + "/" + this.endNewKanji + "]" + (this.soundMorph
? "M" : "");
        }
    }
}

```

```

public RuleInstance()
{
}

public RuleInstance(string perfectMatch)
{
    this.srcWordRomaji = perfectMatch;
    this.srcWordKana =
KanjiTransliterator.kanaTransliterator.Transliterate(perfectMatch
);

    this.srcTag = "any";

    this.startNewKana = "";
    this.startNewKanji = null;

    this.endOrigKana = "";
    this.endNewKana = "";
    this.endNewKanji = null;
}

public List<string> Replace(string dictionaryWord)
{
    List<string> inflectedWords = new List<string>();
    StringBuilder kanaSb = new StringBuilder(startNewKana);
    int soundMorphModifier = this.soundMorph ? 1 : 0;
    string middle = dictionaryWord.Substring(0,
dictionaryWord.Length - this.endOrigKana.Length +
soundMorphModifier);
    kanaSb.Append(middle);

    kanaSb.Append(this.endNewKana.Substring(soundMorphModifier));

    inflectedWords.Add(kanaSb.ToString());

    if(this.startNewKanji != null || this.endNewKanji != null)
    {
        StringBuilder kanjiSb = new StringBuilder();
        if(this.startNewKanji != null)
        {
            kanjiSb.Append(this.startNewKanji);
        }
        else
        {
            kanjiSb.Append(this.startNewKana);
        }
        kanjiSb.Append(middle);
        if(this.endNewKanji != null)
        {
            kanjiSb.Append(this.endNewKanji.Substring(soundMorphModifier));
        }
        else
        {

```

```

        kanjiSb.Append(this.endNewKana.Substring(soundMorphModifier));
    }

    inflectedWords.Add(kanjiSb.ToString());
}

return inflectedWords;
}
}
}
}

```

### B.2.8 StringTransliterator.cs

```

using System;
using System.IO;

namespace GamaIme
{
    public class StringTransliterator : Transliterator<char,
char>
    {
        public StringTransliterator()
            : base(StringTransliterator.PassTrough)
        {
        }
        public void AddRule(string source, string dest)
        {
            this.AddRule(source.ToCharArray(),
dest.ToCharArray());
        }
        public string Transliterate(string source)
        {
            return new
String(this.Transliterate(source.ToCharArray()));
        }
        static char[] PassTrough(char c)
        {
            return new char[] { c };
        }
        // each line contains an input and output separated by a
tab
        // when multiple tabs are present, the first one is the
separator
        // empty lines are ignored
        // using tab as an input is unsupported
        public void ParseRule(string rules)
        {
            using (StringReader sr = new StringReader(rules))
            {
                string inputLine;
                while ((inputLine = sr.ReadLine()) != null)
                {
                    if (inputLine.Length == 0)
                    {

```



```

    {
    }
    public Transliterator(TransliterationPolicy
transliterationPolicy)
    {
        this.transliterationPolicy = transliterationPolicy;
    }
    public Transliterator(NoMatchHandler<S, D>
noMatchHandler)
    {
        this.transliterationPolicy =
TransliterationPolicy.DelegateIfNoMatch;
        this.noMatchHandler = noMatchHandler;
    }
    public Transliterator(D[] defaultOutput,
DefaultIfNoMatchBehavior defaultIfNoMatchBehaviour)
    {
        this.transliterationPolicy =
TransliterationPolicy.DefaultIfNoMatch;
        this.DefaultOutput = defaultOutput;
        this.defaultIfNoMatchBehavior =
defaultIfNoMatchBehaviour;
    }
}

#endregion

#region PROPERTIES

public TransliterationPolicy TransliterationPolicy
{
    get
    {
        return this.transliterationPolicy;
    }
    set
    {
        this.transliterationPolicy = value;
    }
}
public DefaultIfNoMatchBehavior DefaultIfNoMatchBehaviour
{
    get
    {
        return this.defaultIfNoMatchBehavior;
    }
    set
    {
        this.defaultIfNoMatchBehavior = value;
    }
}
public D[] DefaultOutput
{
    get
    {
        return this.defaultOutput;
    }
}

```

```

        set
        {
            if (value == null)
            {
                throw new
ArgumentNullException("defaultOutput");
            }
            this.defaultOutput = value;
        }
    }
    public NoMatchHandler<S, D> NoMatchHandler
    {
        get
        {
            return this.noMatchHandler;
        }
        set
        {
            this.noMatchHandler = value;
        }
    }
}

#endregion

public void AddRule(S[] source, D[] dest)
{
    if (source == null)
    {
        throw new ArgumentNullException("source");
    }
    if (dest == null)
    {
        throw new ArgumentNullException("dest");
    }
    List<Node> current = this.root;
    for (int i = 0; i < source.Length; i++)
    {
        S s = source[i];
        Node found;
        if ((found = current.Find(
            delegate(Node match)
            {
                if (match.source.Equals(s))
                {
                    return true;
                }
                else
                {
                    return false;
                }
            }
        )) != null)
        {
            if (i == source.Length - 1)
            {
                if (found.dest != null)

```

```

        {
            System.Text.StringBuilder sb = new
System.Text.StringBuilder("{}");
            foreach(S sElem in source)
            {
                sb.Append(sElem);
                sb.Append(",");
            }
            if(sb.Length > 1)
            {
                sb.Remove(sb.Length - 1, 1);
            }
            sb.Append("{}");
            throw new ArgumentException("Rule
with source " + sb.ToString() + " already exists.", "source");
        }
        found.dest = dest;
    }
    current = found.next;
}
else
{
    Node newNode;
    if (i == source.Length - 1)
    {
        newNode = new Node(s, dest);
    }
    else
    {
        newNode = new Node(s);
    }
    current.Add(newNode);
    current = newNode.next;
}
}
}
public void AddRule(S source, D[] dest)
{
    this.AddRule(new S[] { source }, dest);
}
public void AddRule(S[] source, D dest)
{
    this.AddRule(source, new D[] { dest });
}
public void AddRule(S source, D dest)
{
    this.AddRule(new S[] { source }, dest);
}
/// <example>
/// Suppose the rules are:
/// a -> 1
/// b -> 2
/// c -> 3
/// ab -> 4
/// default: 0
///

```

```

    /// Then (default behaviour):
    /// abcabba -> 43421
    /// bazzab -> ArgumentException thrown
    ///
    /// Using TransliterationPolicy.IgnoreIfNoMatch:
    /// bazzab -> 214
    ///
    /// Using TransliterationPolicy.DefaultIfNoMatch and
DefaultIfNoMatchBehavior.Normal:
    /// bazzab -> 210004
    ///
    /// Using TransliterationPolicy.DefaultIfNoMatch and
DefaultIfNoMatchBehavior.Condensed:
    /// bazzab -> 2104
    /// </example>
    public D[] Transliterate(S[] source)
    {
        if (this.transliterationPolicy ==
TransliterationPolicy.DelegateIfNoMatch && this.noMatchHandler ==
null)
        {
            throw new NullReferenceException("The
transliteration policy is set to
TransliterationPolicy.DelegateIfNoMatch but there is no delegate
to call. Consider setting the property NoMatchHandler.");
        }
        if (source == null)
        {
            throw new ArgumentNullException("source");
        }
        List<D> dest = new List<D>();
        int i = 0;
        int noMatchCount = 0;
        while (i < source.Length)
        {
            List<Node> nextNodes = this.root;
            Node found;
            Node lastNodeFound = null;
            int searchStartIndex = i;
            while (i < source.Length &&
                (found = nextNodes.Find(delegate(Node node) {
return node.source.Equals(source[i]); }))) != null)
            {
                lastNodeFound = found;
                nextNodes = lastNodeFound.next;
                i++;
            }
            if (lastNodeFound == null || lastNodeFound.dest
== null)
            {
                i = searchStartIndex + 1;
                switch (this.transliterationPolicy)
                {
                    case
TransliterationPolicy.DelegateIfNoMatch:
                        dest.AddRange(this.noMatchHandler(sou

```

```

rce[searchStartIndex]));
        break;
        case
TransliterationPolicy.DefaultIfNoMatch:
            noMatchCount++;
            break;
        case
TransliterationPolicy.IgnoreIfNoMatch:
            break;
        default:
            throw new ArgumentException("Cannot
transliterate using current rule.", "source");
    }
    }
    else
    {
        if (noMatchCount > 0)
        {
            if (this.defaultIfNoMatchBehavior ==
DefaultIfNoMatchBehavior.Normal)
            {
                for (int j = 0; j < noMatchCount; j+
+)
                {
                    dest.AddRange(this.defaultOutput)
;
                }
            }
            else
            {
                dest.AddRange(this.defaultOutput);
            }
            noMatchCount = 0;
        }
        dest.AddRange(lastNodeFound.dest);
    }
}

if (noMatchCount > 0)
{
    if (this.defaultIfNoMatchBehavior ==
DefaultIfNoMatchBehavior.Normal)
    {
        for (int j = 0; j < noMatchCount; j++)
        {
            dest.AddRange(this.defaultOutput);
        }
    }
    else
    {
        dest.AddRange(this.defaultOutput);
    }
}

return dest.ToArray();
}

```

```
class Node
{
    public S source;
    public D[] dest;
    public List<Node> next = new List<Node>(0);
    public Node(S source, D[] dest)
    {
        this.source = source;
        this.dest = dest;
    }
    public Node(S source)
        : this(source, null)
    {
    }
}
}
```

### B.2.10 edict\_tags.txt

```
abbr
adj
adv
adv-n
adj-na
adj-no
adj-pn
adj-s
adj-t
arch
ateji
aux
aux-v
conj
col
exp
ek
fam
fem
gikun
gram
hon
hum
id
int
iK
ik
io
MA
male
m-sl
n
n-adv
n-t
```

```
n-suf  
n-pref  
neg  
neg-v  
num  
obs  
obsc  
oK  
ok  
pol  
pref  
prt  
qv  
sl  
suf  
uK  
uk  
v1  
v5  
v5u  
v5u-s  
v5k  
v5g  
v5s  
v5t  
v5n  
v5b  
v5m  
v5r  
v5k-s  
v5aru  
v5uru  
vi  
vs  
vs-i  
vs-s  
vz  
vk  
vt  
vulg  
X  
any
```

### B.2.11 hiragana\_to\_romaji.txt

```
あ xa  
い xi  
う xu  
え xe  
お xo  
  
っ t'  
  
ゃ xya  
ゅ xyu
```

よ xyo

わ xwa

あ a

い i

う u

え e

お o

か ka

き ki

く ku

け ke

こ ko

きや kya

きゅ kyu

きよ kyo

が ga

ぎ gi

ぐ gu

げ ge

ご go

ぎや gya

ぎゅ gyu

ぎよ gyo

さ sa

し si

す su

せ se

そ so

しゃ sha

しゅ shu

しょ sho

ざ za

じ ji

ず zu

ぜ ze

ぞ zo

じゃ ja

じゅ ju

じょ jo

た ta

ち ti

つ tu

て te

と to

ちや cha  
ちゆ chu  
ちよ cho

だ da  
ぢ di  
づ du  
で de  
ど do

な na  
に ni  
ぬ nu  
ね ne  
の no

にや nya  
にゆ nyu  
によ nyo

は ha  
ひ hi  
ふ fu  
へ he  
ほ ho

ひや hya  
ひゆ hyu  
ひよ hyo

ふあ fa  
ふい fi  
ふえ fe  
ふお fo

ば ba  
び bi  
ぶ bu  
べ be  
ぼ bo

びや bya  
びゆ byu  
びよ byo

ぱ pa  
ぴ pi  
ぷ pu  
ぺ pe  
ぽ po

ぴや pya  
ぴゆ pyu  
ぴよ pyo

ま ma

```

み mi
む mu
め me
も mo

みや mya
みゆ myu
みよ myo

や ya
ゆ yu
よ yo

ら ra
り ri
る ru
れ re
ろ ro

りゃ rya
りゅ ryu
りょ ryo

わ wa
を wo

う vu

ん n'

```

### B.2.12 restriction.txt

```

% format:
% tag1[,tag2,...,tagn]:ending1[,ending2,...,endingn]

% example:
% vs-i,vs-s:suru
% the restriction above means that all vs-i and vs-s must end
with suru

adj:i
v1:iru,eru
v5s:su
v5k:ku
v5k-s:iku,yuku
v5g:gu
v5b:bu
v5t:tu
v5m:mu
v5r:ru
v5aru:aru
v5uru:uru
v5n:nu
v5u,v5u-s:u
vs-i,vs-s:suru

```

```
vz:zuru
vk:kuru
te:te,de
```

### B.2.13 romaji\_to\_hiragana.txt

```
a あ
i い
u う
e え
o お

xa あ
xi い
xu う
xe え
xo お

ka か
ki き
ku く
ke け
ko こ

kka つか
kki つき
kku つく
kke つけ
kko つこ

kya きや
kyi きい
kyu きゆ
kye きえ
kyo きよ

kkya つきや
kkyi つきい
kkyu つきゆ
kkye つきえ
kkyo つきよ

ga が
gi ぎ
gu ぐ
ge げ
go ご

gga っが
ggi っぎ
ggu っぐ
gge っげ
ggo っご
```

gya ぎや  
 gyi ぎい  
 gyu ぎゆ  
 gye ぎえ  
 gyo ぎよ

ggya つぎや  
 ggyi つぎい  
 ggyu つぎゆ  
 ggye つぎえ  
 ggyo つぎよ

sa さ  
 si し  
 su す  
 se せ  
 so そ

ssa つさ  
 ssi つし  
 ssu つす  
 sse つせ  
 sso つそ

sya しや  
 syi しい  
 syu しゆ  
 sye しえ  
 syo しよ

ssya つしや  
 ssyi つしい  
 ssyu つしゆ  
 ssye つしえ  
 ssyo つしよ

sha しや  
 shi し  
 shu しゆ  
 she しえ  
 sho しよ

ssha つしや  
 sshi つし  
 sshu つしゆ  
 sshe つしえ  
 ssho つしよ

za ざ  
 zi じ  
 zu ず  
 ze ぜ  
 zo ぞ

zza つざ  
 zzi つじ

zzu っず  
 zze っぜ  
 zzo っぞ

zya じゃ  
 zyi じい  
 zyu じゆ  
 zye じえ  
 zyo じよ

zzya っじゃ  
 zzyi っじい  
 zzyu っじゆ  
 zzye っじえ  
 zzyo っじよ

ja じゃ  
 ji じ  
 ju じゆ  
 je じえ  
 jo じよ

jja っじゃ  
 jji っじ  
 jju っじゆ  
 jje っじえ  
 jjo っじよ

jya じゃ  
 jyi じい  
 jyu じゆ  
 jye じえ  
 jyo じよ

jjya っじゃ  
 jjyi っじい  
 jjyu っじゆ  
 jjye っじえ  
 jjyo っじよ

ta た  
 ti ち  
 tu っ  
 te て  
 to と

tta った  
 tti っち  
 ttu っつ  
 tte って  
 tto っと

xtu っ

t'っ

tha てや  
 thi てい  
 thu てゆ  
 the てえ  
 tho てよ

ttha ってや  
 tthi ってい  
 tthu ってゆ  
 ttthe ってえ  
 ttho ってよ

tya ちゃ  
 tyi ちい  
 tyu ちゆ  
 tye ちえ  
 tyo ちよ

ttya っちや  
 ttyi っちい  
 ttyu っちゆ  
 ttye っちえ  
 ttyo っちよ

cha ちゃ  
 chi ち  
 chu ちゆ  
 che ちえ  
 cho ちよ

ccha っちや  
 cchi っち  
 cchu っちゆ  
 cche っちえ  
 ccho っちよ

cya ちゃ  
 cyi ちい  
 cyu ちゆ  
 cye ちえ  
 cyo ちよ

ccya っちや  
 ccyi っちい  
 ccyu っちゆ  
 ccye っちえ  
 ccyo っちよ

tsu つ  
 ttsu っつ

da だ  
 di ぢ  
 du づ  
 de で  
 do ど

dda っだ  
ddi っぢ  
ddu っづ  
dde っで  
ddo っど

dya ぢゃ  
dyi ぢい  
dyu ぢゅ  
dye ぢえ  
dyo ぢよ

ddya っぢゃ  
ddyi っぢい  
ddyu っぢゅ  
ddye っぢえ  
ddyo っぢよ

dha でゃ  
dhi でい  
dhu でゅ  
dhe でえ  
dho でよ

ddha っでゃ  
ddhi っでい  
ddhu っでゅ  
ddhe っでえ  
ddho っでよ

na な  
ni に  
nu ぬ  
ne ね  
no の

nya にゃ  
nyi にい  
nyu にゅ  
nye にえ  
nyo によ

ha は  
hi ひ  
hu ふ  
he へ  
ho ほ

hha っは  
hhi っひ  
hhu っふ  
hhe っへ  
hho っほ

hya ひゃ

hyi ひい  
hyu ひゆ  
hye ひえ  
hyo ひよ

hhya つひや  
hhyi つひい  
hhyu つひゆ  
hhye つひえ  
hhyo つひよ

fa ふあ  
fi ふい  
fu ふ  
fe ふえ  
fo ふお

ffa つふあ  
ffi つふい  
ffu つふ  
ffe つふえ  
ffo つふお

fya ふや  
fyi ふい  
fyu ふゆ  
fye ふえ  
fyo ふよ

ffya つふや  
ffyi つふい  
ffyu つふゆ  
ffye つふえ  
ffyo つふよ

ba ば  
bi び  
bu ぶ  
be べ  
bo ぼ

bba つば  
bbi つび  
bbu つぶ  
bbe つべ  
bbo つぼ

bya びや  
byi びい  
byu びゆ  
bye びえ  
byo びよ

bbya つびや  
bbyi つびい  
bbyu つびゆ

bbye つびえ  
bbyo つびよ

pa ぱ  
pi ぴ  
pu ぷ  
pe ぺ  
po ぽ

ppa つぱ  
ppi つぴ  
ppu つぷ  
ppe つぺ  
ppo つぽ

pya ぴゃ  
pyi ぴい  
pyu ぴゅ  
pye ぴえ  
pyo ぴよ

ppya つぴゃ  
ppyi つぴい  
ppyu つぴゅ  
ppye つぴえ  
ppyo つぴよ

ma ま  
mi み  
mu む  
me め  
mo も

mma つま  
mmi つみ  
mmu つむ  
mme つめ  
mmo つも

mya みゃ  
myi みい  
myu みゅ  
mye みえ  
myo みよ

mmya つみゃ  
mmyi つみい  
mmyu つみゅ  
mmye つみえ  
mmyo つみよ

ya や  
yi い  
yu ゆ  
ye いえ  
yo よ

yya つや  
yyi つい  
yyu つゆ  
yye ついえ  
yyo つよ

xya や  
xyu ゆ  
xyo よ

ra ら  
ri り  
ru る  
re れ  
ro ろ

rra つら  
rri つり  
rru つる  
rre つれ  
rro つろ

rya りや  
ryi りい  
ryu りゆ  
rye りえ  
ryo りよ

rrya つりや  
rryi つりい  
rryu つりゆ  
rrye つりえ  
rryo つりよ

wa わ  
wi うい  
wu う  
we うえ  
wo を

xwa わ

wwa つわ  
wwi つうい  
wwu つう  
wwe つうえ  
wwo つを

wha うあ  
whi うい  
whu う  
whe うえ  
who うお

wwha つうあ



KKE ッケ  
KKO ッコ

KYA キヤ  
KYI キイ  
KYU キユ  
KYE キエ  
KYO キョ

KKYA ッキヤ  
KKYI ッキイ  
KKYU ッキユ  
KKYE ッキエ  
KKYO ッキョ

GA ガ  
GI ギ  
GU グ  
GE ゲ  
GO ゴ

GGA ッガ  
GGI ッギ  
GGU ッグ  
GGE ッゲ  
GGO ッゴ

GYA ギヤ  
GYI ギイ  
GYU ギユ  
GYE ギエ  
GYO ギョ

GGYA ッギヤ  
GGYI ッギイ  
GGYU ッギユ  
GGYE ッギエ  
GGYO ッギョ

SA サ  
SI シ  
SU ス  
SE セ  
SO ソ

SSA ッサ  
SSI ッシ  
SSU ッス  
SSE ッセ  
SSO ッソ

SYA シヤ  
SYI シイ  
SYU シユ  
SYE シエ  
SYO ショ

SSYA ツシヤ  
SSYI ツシイ  
SSYU ツシユ  
SSYE ツシエ  
SSYO ツシヨ

SHA シヤ  
SHI シ  
SHU シユ  
SHE シエ  
SHO シヨ

SSHA ツシヤ  
SSHI ツシ  
SSHU ツシユ  
SSHE ツシエ  
SSHO ツシヨ

ZA ザ  
ZI ジ  
ZU ズ  
ZE ゼ  
ZO ズ

ZZA ツザ  
ZZI ツジ  
ZZU ツズ  
ZZE ツゼ  
ZZO ツゾ

ZYA ジャ  
ZYI ジイ  
ZYU ジユ  
ZYE ジエ  
ZYO ジヨ

ZZYA ツジャ  
ZZYI ツジイ  
ZZYU ツジユ  
ZZYE ツジエ  
ZZYO ツジヨ

JA ジャ  
JI ジ  
JU ジュ  
JE ジエ  
JO ジヨ

JJA ツジャ  
JJI ツジ  
JJU ツジユ  
JJE ツジエ  
JJO ツジヨ

JYA ジャ

JYI ジイ  
JYU ジュ  
JYE ジェ  
JYO ジョ

JJYA ツジャ  
JJYI ツジイ  
JJYU ツジュ  
JJYE ツジェ  
JJYO ツジョ

TA タ  
TI チ  
TU ツ  
TE テ  
TO ト

TTA ッタ  
TTI ッチ  
TTU ッツ  
TTE ッテ  
TTO ット

XTU ツ

THA テヤ  
THI テイ  
THU テユ  
THE テェ  
THO テョ

TTHA ッテヤ  
TTHI ッテイ  
TTHU ッテユ  
TTHE ッテェ  
TTHO ッテョ

TYA チャ  
TYI チイ  
TYU チュ  
TYE チェ  
TYO チョ

TTYA ッチャ  
TTYI ッチイ  
TTYU ッチュ  
TTYE ッチェ  
TTYO ッチョ

CHA チャ  
CHI チ  
CHU チュ  
CHE チェ  
CHO チョ

CCHA ッチャ

CCHI ッチ  
CCHU ッチュ  
CCHE ッチエ  
CCHO ッチヨ

CYA チャ  
CYI チイ  
CYU チュ  
CYE チェ  
CYO チョ

CCYA ッチャ  
CCYI ッチイ  
CCYU ッチュ  
CCYE ッチエ  
CCYO ッチヨ

TSU ツ  
TTSU ッツ

DAダ  
DIヂ  
DUヅ  
DEデ  
DOド

DDA ッダ  
DDI ッヂ  
DDU ッヅ  
DDE ッデ  
DDO ッド

DYA チャ  
DYI チイ  
DYU チュ  
DYE チェ  
DYO チョ

DDYA ッチャ  
DDYI ッチイ  
DDYU ッチュ  
DDYE ッチエ  
DDYO ッチヨ

DHA デヤ  
DHI デイ  
DHU デユ  
DHE デェ  
DHO デョ

DDHA ッデヤ  
DDHI ッデイ  
DDHU ッデユ  
DDHE ッデェ  
DDHO ッデョ

NAナ  
NIニ  
NUヌ  
NEネ  
NOノ

NYA ニヤ  
NYI ニイ  
NYU ニユ  
NYE ニエ  
NYO ニヨ

HAハ  
HIヒ  
HUフ  
HEへ  
HOホ

HHA ツハ  
HHI ツヒ  
HHU ツフ  
HHE ツへ  
HHO ツホ

HYA ヒヤ  
HYI ヒイ  
HYU ヒユ  
HYE ヒエ  
HYO ヒヨ

HHYA ツヒヤ  
HHYI ツヒイ  
HHYU ツヒユ  
HHYE ツヒエ  
HHYO ツヒヨ

FAファ  
FIファイ  
FUフ  
FEフェ  
FOフォ

FFA ツファ  
FFI ツファイ  
FFU ツフ  
FFE ツフェ  
FFO ツフォ

FYA ファ  
FYI ファイ  
FYU フユ  
FYE フェ  
FYO フヨ

FFYA ツフヤ  
FFYI ツファイ

FFYU ッフユ  
FFYE ッフエ  
FFYO ッフヨ

BAバ  
BIビ  
BUブ  
BEべ  
BOボ

BBA ッバ  
BBI ッビ  
BBU ッブ  
BBE ッべ  
BBO ッボ

BYA ビャ  
BYI ビイ  
BYU ビュ  
BYE ビエ  
BYO ビョ

BBYA ッビャ  
BBYI ッビイ  
BBYU ッビュ  
BBYE ッビエ  
BBYO ッビョ

PAパ  
PIピ  
PUプ  
PEぺ  
POポ

PPA ッパ  
PPI ッピ  
PPU ップ  
PPE ッぺ  
PPO ッポ

PYA ピャ  
PYI ピイ  
PYU ピュ  
PYE ピエ  
PYO ピョ

PPYA ッピャ  
PPYI ッピイ  
PPYU ッピュ  
PPYE ッピエ  
PPYO ッピョ

MAマ  
MIミ  
MUム  
MEメ

MOモ

MMA ツマ

MMI ツミ

MMU ツム

MME ツメ

MMO ツモ

MYA ミヤ

MYI ミイ

MYU ミユ

MYE ミエ

MYO ミヨ

MMYA ツミヤ

MMYI ツミイ

MMYU ツミユ

MMYE ツミエ

MMYO ツミヨ

YAヤ

YIイ

YUユ

YEイエ

YOヨ

YYA ツヤ

YYI ツイ

YYU ツユ

YYE ツイエ

YYO ツヨ

XYA ヤ

XYU ユ

XYO ヨ

RAラ

RIリ

RUル

REレ

ROロ

RRA ツラ

RRI ツリ

RRU ツル

RRE ツレ

RRO ツロ

RYA リヤ

RYI リイ

RYU リユ

RYE リエ

RYO リヨ

RRYA ツリヤ

RRYI ツリイ

RRYU ッリユ  
RRYE ッリエ  
RRYO ッリヨ

WA ワ  
WI ウイ  
WU ウ  
WE ウェ  
WO ワ

XWA ワ

WWA ッワ  
WWI ッウイ  
WWU ッウ  
WWE ッウエ  
WWO ッワ

WHA ウア  
WHI ウイ  
WHU ウ  
WHE ウェ  
WHO ウオ

WWHA ッウア  
WWHI ッウイ  
WWHU ッウ  
WWHE ッウエ  
WWHO ッウオ

VA ヴァ  
VI ヴィ  
VU ヴ  
VE ヴェ  
VO ヴォ

VVA ッヴァ  
VVI ッヴィ  
VVU ッヴ  
VVE ッヴェ  
VVO ッヴォ

VYA ヴャ  
VYI ヴィ  
VYU ヴュ  
VYE ヴェ  
VYO ヴョ

VVYA ッヴァ  
VVYI ッヴィ  
VVYU ッヴュ  
VVYE ッヴェ  
VVYO ッヴォ

N ン  
N' ン

- -

### B.2.15 rules\_romaji.txt

```
% format:
% tag1[,tag2,...,tagN]/source_pattern/dest_pattern/newtag

% example:
% n,adj-na/*/*janai/neg
% the rule above means that if a word is a noun (n) or a na-
adjective (adj-na),
% it can receive the ending "janai" to make it negative
% (a new tag "neg" will be applied)

% some sound change not indicated in romanization!
% NICE:
% a, i, u, e, o
% ka, ki, ku, ke, ko, ga, gi, gu, ge, go
% na, ni, nu, ne, no
% ma, mi, mu, me, mo
% ya, yu, yo
% wa, wo
% n'
% CAREFUL:
% sa, SI, su, se, so, za, JI, zu, ze, zo
% ta, TI, TU, te, to, da, DI, du, de, do
% ha, hi, FU, he, ho, ba, bi, bu, be, bo, pa, pi, pu, pe, po
% っ is t'

% uncomment below for early termination
%TERMINATE

% state of being
n,adj-na/*/*da/decl
n,adj-na/*/*janai/neg
n,adj-na/*/*dat'ta/past
neg/*i/*kat'ta/past

% particles
any/*/*ha/particle
any/*/*ga/particle
any/*/*mo/particle

% adjectives
% na is now made any
% n,adj-na/*/*na/adj
adj/*i/*kunai/neg
adj/*i/*kat'ta/past

% negative verbs
v1/*ru/*nai/neg
v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n/*u/*anai/neg
v5u,v5u-s/*u/*wanai/neg
vs-i,vs-s/*(su)ru/*(si)nai/neg
```

```

vz/*zuru/*jinai/neg
vk/*(ku)ru/*(ko)nai/neg

% p58: no rule for aru -> nai

% past verbs
v1/*ru/*ta/past
v5s/*su/*sita/past
v5k/*ku/*ita/past
v5g/*gu/*ida/past
v5m/*mu/*n'da/past
v5b/*bu/*n'da/past
v5n/*nu/*n'da/past
v5r,v5aru/*ru/*t'ta/past
v5u/*u/*t'ta/past
v5u-s/*/*ta/past
v5t/*tu/*t'ta/past
vs-i,vs-s/*(su)ru/*(si)ta/past
vz/*zuru/*jita/past
vk/*(ku)ru/*(ki)ta/past
v5k-s/*ku/*t'ta/past
v5k-s/*(yu)ku/*(yu)ita/past

% more particles
any/*/*wo/particle
any/*/*ni/particle,adv
% adj-na,n/*/*ni/adv
any/*/*he/particle
any/*/*de/particle,te
% n,adj-na/*/*de/te
n,adj-na,adv/*/*made/particle

% yet more particles
any/*/*no/particle,n
any/*/*n'/particle,n
any/*/*to/particle,to,vs
any/*/*ya/particle,ya
any/*/*toka/particle,toka

% adverb
adj/*i/*ku/adv

% gobi
any/*/*ne/gobi
any/*/*yo/gobi
%any/*/*yone/gobi
%the above currently can be representation of
%the preceding two rules

% stem
v1/*ru/*/stem,n
v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n,v5u,v5u-
s/*u/*i/stem,n
vs-i,vs-s/*(su)ru/*(si)/stem,n
vz/*zuru/*ji/stem,n
vk/*(ku)ru/*(ki)/stem,n

```

```

% polite forms
stem/*/*masu/masu
masu/*su/*sen'/neg
masu/*su/*sita/past
neg/*sen'/*sen'desita/past

any/*/*desu/desu
any/*/*desita/desu,past

neg/*nai/*arimasen'/masu

% p94: people omitted

any/*/*ka/q

adj,neg/*i/*kute/te

past/*ta/*te/te
past/*da/*de/te

any/*/*kara/kara
any/*/*noni/noni
any/*/*kedo/kedo
any/*/*si/si
past/*/*ri/vs

te/*/*iru/v1
te/*/*ru/v1
te/*/*aru/v5r
te/*/*oku/v5k
te/*/*(i|行)ku/v5k-s
te/*/*(yu|行)ku/v5k-s
te/*/*ku/v5k-s
te/*/*(ku|来)ru/vk

% potential form
v1/*ru/*rareru/pot
v1/*ru/*reru/pot
v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n,v5u,v5u-
s/*u/*eru/v1,pot
vk/*(ku)ru/*(ko)rareru/v1,pot
vk/*(ku)ru/*(ko)reru/v1,pot
vs-i,vs-s/*suru/*(deki|出来)ru/v1,pot
vz/*zuru/*(deki|出来)ru/v1,pot

adv/*/*naru/v5r
adv/*/*suru/vs-i
any/*/*you/n

particle/*/*(i|行)ku/v5k-s
particle/*/*(ku|来)ru/vk

% conditional
any/*/*nara/cond
any/*/*naraba/cond

```

```

v1,v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n,v5u,v5u-s,vs-i,vs-
s,vz,vk/*u/*eba/cond
neg,adj/*i/*kereba/cond
past/*/*ra/cond
past/*/*raba/cond

% must do (nor not)
te/*te/*cha/must
te/*de/*ja/must
cond/*kereba/*kya/must

% desire
stem/*/*tai/adj
te/*/*(ho|欲)sii/adj

% suggestions
v1/*ru/*you/n
v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n,v5u,v5u-s/*u/*ou/n
vs-i,vs-s/*(su)ru/*(si)you/n
vz/*zuru/*jiiyou/n
vk/*(ku)ru/*(ko)you/n
stem/*/*mashou/n
cond/*ba/*badou/n
cond/*tara/*taradou/n

% quote
to/*/*(i|言)u/v5u
to/*/*(omo|思)u/v5u
% to kiku, kangaeru, and others omitted
any/*/*t'te/quote

% trying things out
te/*/*(mi|見)ru/v1

% giving and receiving
% wo-ageru etc omitted
te/*/*(ku|呉)reru/v1
te/*/*(mora|貰)u/v5u

% request
te/*/*kudasai/req
te/*/*choudai/req
stem/*/*nasai/req

% command
v1/*ru/*ro/comm
v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n,v5u,v5u-
s,masu/*u/*e/comm
vs-i,vs-s/*(su)ru/*(si)ro/comm
vz/*zuru/*jiiro/comm
vk/*(ku)ru/*(ko)i/comm

% na is now made any
%v1,v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n,v5u,v5u-s,vs-
i,vs-s,vz,vk/*/*na/comm

```

```

% number and counter skipped

% more gobi
any/**sa/gobi
any/**na/gobi
any/**wa/gobi
any/**zo/gobi
any/**ze/gobi
any/**kasira/gobi

% probably many skipped

% causative and passive forms

%% causative normal
v1/*ru/*saseru/caus
v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n/*u/*aseru/v1,caus
v5u,v5u-s/*u/*waseru/v1,caus
vs-i,vs-s/*(su)ru/*(sa)seru/v1,caus
vk/*(ku)ru/*(ko)saseru/v1,caus
%vz/*zuru/*jinai/neg

%% causative shortened
v1/*ru/*sasu/caus,v5s
v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n/*u/*asu/caus,v5s
v5s/*u/*asu/caus
v5u,v5u-s/*u/*wasu/caus,v5s
vs-i,vs-s/*(su)ru/*(sa)su/caus,v5s
vk/*(ku)ru/*(ko)sasu/caus,v5s
%vz/*zuru/*jinai/neg

%% passive
v1/*u/*areru/pass
v5s,v5k,v5k-s,v5g,v5b,v5t,v5m,v5r,v5aru,v5n/*u/*areru/v1,pass
v5u,v5u-s/*u/*wareru/v1,pass
vs-i,vs-s/*(su)ru/*(sa)reru/v1,pass
vk/*(ku)ru/*(ko)rareru/v1,pass
%vz/*zuru/*jinai/neg

% honorific and humble forms
v5aru/*ru/*imasu/masu
stem/**suru/vs-i
stem/**kudasai/req
masu/*aimasu/*ai/comm

%
% below, just for testing...
%

vs/**suru/vs-i

% should probably make this to any
n,vs,stem/*/(o|御)*/hon
n,vs,stem/*/(go|御)*/hon
n,vs,stem/*/(dai|大)*/dai
n,vs,stem/*/(sin'|新)*/shin

```

```
n,vs,stem/*/kono*/kono
n,vs,stem/*/sono*/sono
n,vs,stem/*/ano*/ano
n,vs,stem/*/dono*/dono
any/*/deshou/deshou
```

### B.3 transliterator.aspx

```
<%@ Page Language="C#" Debug="True" %>
<%@ Import Namespace="GamaIme" %>
<%
    string[] results;

    HiraganaToRomajiTransliterator romajiTrans = new
HiraganaToRomajiTransliterator();
    KanjiTransliterator t = new
KanjiTransliterator(MapPath("rules_romaji.txt"),
MapPath("restriction.txt"), MapPath("edict_tags.txt"),
MapPath("edict.db3"));

    try
    {
        results =
t.Transliterate(romajiTrans.Transliterate(Request.QueryString["hi
ragana"]));
    }
    catch(Exception e)
    {
        results = new string[3];
        results[0] = Request.QueryString["hiragana"];
        results[1] = "INTERNAL ERROR";
        results[2] = e.ToString();
    }

    Response.Write(Helper.JoinArray(results, ";"));
%>
```

### B.4 Aplikasi Web IME

#### B.4.1 ime.htm

```
<html>
  <head>
    <title>Gama Japanese IME</title>
    <meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
    <script type="text/javascript" src="misc.js"></script>
    <script type="text/javascript" src="events.js"></script>
    <script type="text/javascript" src="keyboard.js"></script>
    <script type="text/javascript" src="ime.js"></script>
    <script type="text/javascript"
src="transliterator.js"></script>
    <script type="text/javascript">
      addEvent(window, 'load', function() {
```

```

    });
  </script>

  <link rel="stylesheet" href="popup.css"></link>
</head>
<body>
  <script type="text/javascript">
    writeIme('IME', 500, 200);
  </script>
  <div id="debug"></div>
</body>
</html>

```

### B.4.2 ime.js

```

var detect = {
  ua: navigator.userAgent.toLowerCase(),
  browser: null,
  explorer: null,
  opera: null,
  firefox: null
};

if(detect.ua.indexOf("msie") != -1)
{
  detect.explorer = true;
  detect.browser = 'explorer';
}
else if(detect.ua.indexOf("opera") != -1)
{
  detect.opera = true;
  detect.browser = 'opera';
}
else // assume firefox without detecting 'gecko'
{
  detect.firefox = true;
  detect.browser = 'firefox';
}

if(typeof XMLHttpRequest == 'undefined')
{
  XMLHttpRequest = function()
  {
    return new ActiveXObject(
      navigator.userAgent.indexOf('MSIE 5') >= 0 ?
      'Microsoft.XMLHTTP' : 'Msxml2.XMLHTTP'
    );
  }
}

var ime = {
  on: false,
  cache: new Object(),
  charTable: new Object(), // initialized on writeIme
  keyCode: null,

```

```
requestId: 0, // to track the last AJAX request
// easy DOM access
id: null,
elem: null,
window: null,
doc: null,
body: null
};

ime.keydown = function(event)
{
    ime.keyCode = event.keyCode;
};

ime.keypress = function(event)
{
    var parsedKey;
    if(ime.charTable.special[ime.keyCode])
    {
        parsedKey = ime.charTable.special[ime.keyCode];
    }
    else if(ime.charTable.printable[ime.keyCode])
    {
        if(event.shiftKey)
        {
            parsedKey =
ime.charTable.printable[ime.keyCode].substring(1, 2);
        }
        else
        {
            parsedKey =
ime.charTable.printable[ime.keyCode].substring(0, 1);
        }
    }
    else // alphanumeric
    {
        parsedKey = String.fromCharCode(ime.keyCode); // the 'a' key
will be capital 'A' regardless of shift/caps lock
        var shiftOn = event.shiftKey;

        if(!shiftOn)
        {
            parsedKey = parsedKey.toLowerCase();
        }
    }

    ime.surpressKeypressDefault = false; // you can turn on in
internalKeypress
    ime.internalKeypress(parsedKey, event);

    if(ime.surpressKeypressDefault)
    {
        event.preventDefault();
        event.stopPropagation();
    }
};
```

```

ime.internalKeypress = function(key, event)
{
  if(ime.on)
  {
    ime.surpressKeypressDefault = true;
    if(imePopup.mode == 0) // we are in the textbox
    {
      if(event.ctrlKey) // can be ctrl+c, ctrl+v, etc
      {
        ime.surpressKeypressDefault = false;
      }
      else if(key == ' ') // insert fullwidth space instead of
normal
      {
        insertHTML(fullwidthTrans.transliterate(key));
      }
      else if(key == 'enter') // insert newline
      {
        if(detect.opera && false)
        {
          // after using insertHTML before (e.g., to insert 'あ
'),
          // the Enter key's default action (e.g., inserting a
newline)
          // won't be triggered.
          // Therefore, we must insert newline manually
insertHTML('<br><span id="operaHack">_</span><span>
</span>');
          remove(id('operaHack'));
        }
        else
        {
          ime.surpressKeypressDefault = false;
        }
      }
      else if(key.length == 1) // show popup with 'a' displayed
as 'あ' etc.
      {
        imePopup.show(key);
      }
      else // probably up, down, etc
      {
        ime.surpressKeypressDefault = false;
      }
    }
    else // we are in the popup
    {
      if(key == 'enter')
      {
        if(imePopup.mode == 3)
        {
          imePopup.updateCachePriority();
        }
        imePopup.commit();
      }
    }
  }
}

```

```

        else if(key == 'f6')
        {

imePopup.switchToSelectedMode(hiraganaTrans.transliterate(imePopu
p.romaji));
        }
        else if(key == 'f7')
        {

imePopup.switchToSelectedMode(katakanaTrans.transliterate(imePopu
p.romaji));
        }
        else if(key == 'f8')
        {

imePopup.switchToSelectedMode(halfwidthTrans.transliterate(imePop
up.romaji));
        }
        else if(key == 'f9')
        {

imePopup.switchToSelectedMode(fullwidthTrans.transliterate(imePop
up.romaji));
        }
        else if(key == 'f10')
        {
            imePopup.switchToSelectedMode(imePopup.romaji);
        }
        else if(key == ' ')
        {
            if(imePopup.mode == 1 || imePopup.mode == 2)
            {
                if(ime.cache[imePopup.romaji])
                {
                    imePopup.switchToSelectingMode();
                }
                else
                {
                    ime.requestTransliteration(imePopup.romaji);
                }
            }
            else if(imePopup.mode == 3)
            {
                imePopup.setHighlighted(imePopup.highlighted + 1);
                imePopup.showChoices();
            }
        }
        else if(key == 'up')
        {
            if(imePopup.mode == 3)
            {
                imePopup.setHighlighted(imePopup.highlighted - 1);
            }
        }
    }
}

```

```

        imePopup.showChoices();
    }
}
else if(key == 'down')
{
    if(imePopup.mode == 3)
    {
        imePopup.setHighlighted(imePopup.highlighted + 1);
        imePopup.showChoices();
    }
}
else if(key == 'backspace')
{
    if(imePopup.mode == 1)
    {
        imePopup.setRomaji(imePopup.romaji.substring(0,
imePopup.romaji.length - 1));
        if(imePopup.romaji.length == 0)
        {
            imePopup.hide();
            ime.window.focus();
        }
    }
    else
    {
        imePopup.switchToTypingMode();
    }
}
else if(key == 'pageup' &&
getStyle(imePopup.choicesBox, 'display') != 'none')
{
    // select the topmost option of previous page
    var newHighlighted = Math.floor(imePopup.highlighted /
imePopup.choicePerPage) * imePopup.choicePerPage -
imePopup.choicePerPage;
    imePopup.setHighlighted(newHighlighted);
}
else if(key == 'pagedown' &&
getStyle(imePopup.choicesBox, 'display') != 'none')
{
    // select the topmost option of next page
    var newHighlighted = Math.floor(imePopup.highlighted /
imePopup.choicePerPage) * imePopup.choicePerPage +
imePopup.choicePerPage;
    imePopup.setHighlighted(newHighlighted);
}
else if(/[1-9]/.test(key) && getStyle(imePopup.choicesBox,
'display') != 'none')
{
    imePopup.selectChoice(Number(key));
}
else if(key == 'esc')
{
    if(imePopup.mode == 1)
    {
        imePopup.hide();
    }
}

```

```

        ime.window.focus();
    }
    else
    {
        imePopup.switchToTypingMode();
    }
}
else if(key.length == 1)
{
    if(imePopup.mode == 1)
    {
        imePopup.setRomaji(imePopup.romaji + key);
    }
    else if(imePopup.mode == 2)
    {
        imePopup.commit(key);
    }
    else if(imePopup.mode == 3)
    {
        imePopup.updateCachePriority()
        imePopup.commit(key);
    }
}
}
};

ime.requestTransliteration = function(romaji)
{
    var xml = new XMLHttpRequest();
    var requestId = ++ime.requestId;
    xml.open('GET', 'transliterator.aspx?hiragana=' +
encodeURIComponent(hiraganaTrans.transliterate(romaji)), true);
    xml.onreadystatechange = function(){
        if(xml.readyState == 4)
        {
            // check whether the HTTP response is OK
            // any status in 2xx range is good
            // 304 means 'not modified', or from the browser's cache
            if(xml.status && ((xml.status >= 200 && xml.status < 300)
||
            xml.status == 304))
            {
                // data has arrived correctly
                // fill the cache without caring whether the user
                // still wants this data
                // this is because the data might be useful for future
input
                if(typeof ime.cache[romaji] == 'undefined')
                {
                    ime.cache[romaji] = xml.responseText.split(';');
                }

                // check whether this is the last request
                // because while waiting, the user might change the
romaji

```

```

        // and make another request
        // also check whether the user is still in loding mode
        if(requestId == ime.requestId &&
            imePopup.getModeName() == 'loading')
        {
            imePopup.switchToSelectingMode();
        }
    }
    // try another request, with delay to prevent clogging the
system
    // in case of instant failure (e.g., hotspot not logged
in)
    else
    {
        setTimeout(function(){
            ime.requestTransliteration(romaji)
        }, 1000);
    }
    xml = null;
}
};

if(detect.firefox)
{
    xml.send(null);
}
else
{
    xml.send();
}
imePopup.switchToLoadingMode(requestId);
}

var imePopup = {
    // consts
    choicePerPage: 9,
    modeNames: ['hidden', 'typing', 'selected', 'selecting',
'loading'],
    // normal vars
    choices: null, // the array of string from which the user can
choose from the popup
    highlighted: -1, // the selected index in choices
    mode: 0, // 0 means hidden, 1 means typing, 2 means selected, 3
means selecting (choices shown), 4 means loading (waiting for
AJAX response)
    romaji: '', // the unmodified user input, e.g., watashi
    converted: '', // the converted user input, e.g., わたし
    loadingBoxDisplayer: null, // the timer that displays the
loading box
    // easy DOM access
    elem: null,
    outputBox: null,
    loadingBox: null,
    choicesBox: null,
    firstChoiceRow: null
};

```

```

imePopup.init = function()
{
  this.outputBox = descendant(this.elem, 'popupConverted');
  this.choicesBox = descendant(this.elem, 'popupTable');
  this.loadingBox = descendant(this.elem, 'popupLoading');
  this.firstChoiceRow = descendant(this.elem,
'popupTableFirstRow');
}

imePopup.switchToSelectingMode = function()
{
  if(imePopup.loadingBoxDisplayer)
  {
    clearTimeout(imePopup.loadingBoxDisplayer);
  }
  imePopup.mode = 3;
  imePopup.hideLoadingBox();
  imePopup.choices = ime.cache[imePopup.romaji];
  imePopup.setHighlighted(0);
  this.outputBox.style.borderBottom = 'solid black 1px';
}

imePopup.switchToLoadingMode = function(requestId)
{
  imePopup.mode = 4;
  // displays the loading box after a certain period
  imePopup.loadingBoxDisplayer = setTimeout(function(){
    if(ime.requestId == requestId && imePopup.getModeName() ==
'loading')
    {
      imePopup.showLoadingBox();
    }
  }, 500);
}

imePopup.getModeName = function()
{
  return this.modeNames[this.mode];
}

// input is 1 to this.choicePerPage
imePopup.selectChoice = function(num)
{
  var topChoiceIndex = Math.floor(this.highlighted /
this.choicePerPage) * this.choicePerPage;
  var newIndex = topChoiceIndex + num - 1;
  if(newIndex < this.choices.length)
  {
    this.setHighlighted(newIndex);
    this.hideChoices();
  }
}

imePopup.showChoices = function()
{

```

```

    this.choicesBox.style.display = 'block';
}

imePopup.hideChoices = function()
{
    this.choicesBox.style.display = 'none';
}

// input is 0 to this.choices.length - 1
// when out of range it will be wrapped
imePopup.setHighlighted = function(index)
{
    if(index < 0)
    {
        index = imePopup.choices.length - 1;
    }
    else if(index >= imePopup.choices.length)
    {
        index = 0;
    }
    this.highlighted = index;
    var topIndex = Math.floor(index / imePopup.choicePerPage) *
imePopup.choicePerPage;

    var rowNode = this.firstChoiceRow;

    for(var i = 0; i < imePopup.choicePerPage; i++)
    {
        var currentIndex = topIndex + i;

        if(currentIndex >= imePopup.choices.length)
        {
            first(rowNode).className = 'popupNumberDisabled';
            last(rowNode).className = 'popupChoiceDisabled';
            last(rowNode).innerHTML = '';
        }
        else
        {
            first(rowNode).className = 'popupNumber';
            last(rowNode).innerHTML = imePopup.choices[currentIndex];
            if(currentIndex == index)
            {
                last(rowNode).className = 'popupChoiceSelected';
            }
            else
            {
                last(rowNode).className = 'popupChoice';
            }
        }
        rowNode = next(rowNode);
    }

    first(rowNode).innerHTML = (this.highlighted + 1) + '/' +
this.choices.length;

    this.setConverted(this.choices[this.highlighted]);
}

```

```
}

// modifies the cache so that the selected conversion is pushed
// to the top of choices
imePopup.updateCachePriority = function()
{
    var prevCache = ime.cache[imePopup.romaji];
    ime.cache[imePopup.romaji] = [imePopup.converted];
    for(var i = 0; i < prevCache.length; i++)
    {
        if(prevCache[i] != imePopup.converted)
        {
            ime.cache[imePopup.romaji].push(prevCache[i]);
        }
    }
}

imePopup.setConverted = function(str)
{
    this.converted = str;
    this.outputBox.innerHTML = str;
}

// changes the romaji and updates the outputBox to display the
// hiragana transliteration
imePopup.setRomaji = function(str)
{
    this.romaji = str;
    this.setConverted(hiraganaTrans.transliterate(str));
}

imePopup.switchToTypingMode = function()
{
    this.mode = 1;
    this.setRomaji(imePopup.romaji);
    this.outputBox.style.borderBottom = 'dashed black 1px';
    this.hideLoadingBox();
    this.hideChoices();
}

imePopup.switchToSelectedMode = function(str)
{
    this.mode = 2;
    this.setConverted(str);
    this.outputBox.style.borderBottom = 'solid black 1px';
    this.hideChoices();
    this.hideLoadingBox();
}

imePopup.hideLoadingBox = function()
{
    this.loadingBox.style.display = 'none';
}

imePopup.showLoadingBox = function()
{
```

```

    this.loadingBox.style.display = 'block';
}

function writeIme(name, width, height)
{
    ime.id = name;
    document.writeln('IME is <span id="imeStatus">off</span>.  

<input id="imeButton" type="button" value="Turn IME On"  

onclick="switchIme()"></input><br /><br />');
    document.writeln('<iframe id="' + ime.id + '" name="' + ime.id  

+ '" src="blank.htm" style="width: ' + width + 'px; height: ' +  

height + 'px;"></iframe>');

    var html = '<html>\n';
    html += '  <head>\n';
    html += '    <style>\n';
    html += '      body{\n';
    html += '        background: white;\n';
    html += '        color: black;\n';
    html += '      }\n';
    html += '    </style>\n';
    html += '  </head>\n';
    html += '  <body>\n';
    html += '  </body>\n';
    html += '</html>';

    ime.elem = id(ime.id);
    ime.window = frames[ime.id];
    ime.doc = ime.window.document;
    ime.body = ime.doc.body;

    // if these lines are omitted, setting designMode has no effect
    ime.doc.open();
    ime.doc.write(html);
    ime.doc.close();

    ime.doc.designMode = 'On';

    ime.charTable.special = charTable.special[detect.browser];
    ime.charTable.printable = charTable.printable[detect.browser];

    // for case where the focus is on the iFrame document
    addEvent(ime.doc, 'keydown', ime.keydown);
    if(detect.opera)
    {
        addEvent(ime.doc, 'keypress', ime.keypress);
    }
    else
    {
        addEvent(ime.doc, 'keydown', ime.keypress);
    }

    addEvent(ime.doc, 'mouseup', function(event){
        imePopup.hide();
    });
}

```

```

// init imePopup

imePopup.elem = checkElem('<div id="popup" style="position:
absolute;">\n' +
'    <div><span id="popupConverted" style="background: white;
border-bottom: dashed black 1px;">わた</span></div>\n' +
'    <div id="popupLoading" style="display: none"><span
style="background: rgb(220,220,220); color: black; padding-left:
5px; padding-right: 5px; border: solid black
3px;">Loading...</span></div>\n' +
'    <table id="popupTable" class="popupTable"
cellspacing="0" style="background: rgb(220,220,220); border:
solid black 3px; position: relative; left: -22px; display:
none;">\n' +
'        <tr id="popupTableFirstRow"><td
class="popupNumber">1</td><td
class="popupChoice">bla</td></tr>\n' +
'        <tr><td class="popupNumber">2</td><td
class="popupChoice">bla bla bla</td></tr>\n' +
'        <tr><td class="popupNumber">3</td><td
class="popupChoice">bla</td></tr>\n' +
'        <tr><td class="popupNumber">4</td><td
class="popupChoice">bla</td></tr>\n' +
'        <tr><td class="popupNumber">5</td><td
class="popupChoiceSelected">bla</td></tr>\n' +
'        <tr><td class="popupNumber">6</td><td
class="popupChoice">bla</td></tr>\n' +
'        <tr><td class="popupNumberDisabled">7</td><td
class="popupChoiceDisabled"></td></tr>\n' +
'        <tr><td class="popupNumberDisabled">8</td><td
class="popupChoicDisablede"></td></tr>\n' +
'        <tr><td class="popupNumberDisabled">9</td><td
class="popupChoiceDisabled"></td></tr>\n' +
'        <tr><td colspan="2" style="border-top: ridge; text-
align: right">3/7</td></tr>\n' +
'    </table>\n' +
'    </div>') [0];

imePopup.init();

addEvent(imePopup.outputBox, 'click', function(event){
    event.stopPropagation();
    return false;
});

addEvent(imePopup.loadingBox, 'click', function(event){
    event.stopPropagation();
    return false;
});

addEvent(imePopup.choicesBox, 'click', function(event){
    event.stopPropagation();
    return false;
});

```

```

var row = imePopup.firstChoiceRow;
var i = 1;
do
{
    (function(){
        var curChoice = i;
        addEvent(row, 'click', function(event){
            imePopup.selectChoice(curChoice);
            event.stopPropagation();
            return false;
        });
    })();
    i++;
    row = next(row);
}while(i <= imePopup.choicePerPage);

if(detect.explorer)
{
    addEvent(document, 'click', function(){imePopup.hide()});

    // for case where the focus is on the popup
    addEvent(imePopup.elem, 'keydown', ime.keydown);
    addEvent(imePopup.elem, 'keydown', ime.keypress);
}
else
{
    addEvent(window, 'click', function(){imePopup.hide()});

    // for case where the focus is NOT on the iFrame (and not on
the popup either, as popups (div) can't get focus on Mozilla and
Opera
    addEvent(window, 'keydown', function(event)    {
        if(imePopup.mode != 0)
        {
            ime.keydown(event);
        }
    });

    if(detect.opera)
    {
        addEvent(window, 'keypress', function(event){
            if(imePopup.mode != 0)
            {
                ime.keypress(event);
            }
        });
    }
    else // firefox
    {
        addEvent(window, 'keydown', function(event){
            if(imePopup.mode != 0)
            {
                ime.keypress(event);
            }
        });
    }
}

```

```
    }  
  }  
  
function blurIme()  
{  
  if(!blurIme.dummyInput)  
  {  
    blurIme.dummyInput = checkElem('<input type="text" id="dummyInput" style="width:0px;"></input>')[0];  
  }  
  append(document.body, blurIme.dummyInput);  
  blurIme.dummyInput.focus();  
  remove(blurIme.dummyInput);  
}  
  
function insertHTML(html) {  
  ime.window.focus();  
  if (detect.explorer) {  
    ime.doc.selection.createRange().pasteHTML(html);  
  } else {  
    ime.doc.execCommand('insertHTML', false, html);  
  }  
}  
  
imePopup.show = function(key)  
{  
  // TODO: check whether there is a code that calls this function  
  without arguments  
  var c = getImeCursorLoc();  
  blurIme();  
  this.elem.style.left = c.x + 'px';  
  this.elem.style.top = c.y + 'px';  
  append(document.body, this.elem);  
  if(key)  
  {  
    this.setRomaji(key);  
    this.mode = 1;  
    this.outputBox.style.borderBottom = 'dashed black 1px';  
  }  
  if(detect.explorer)  
  {  
    this.elem.focus();  
  }  
}  
  
//var x = 0; // todo: delete this x thing  
imePopup.hide = function()  
{  
  //debug('hide called: ' + (x++));  
  if(this.mode != 0)  
  {  
    this.mode = 0;  
    this.hideLoadingBox();  
    this.hideChoices();  
    remove(this.elem);  
  }  
}
```

```

    }
}

function getImeCursorLoc()
{
    insertHTML('<img id="dummyCursor" style="width: 0px; height: 0.9em;" />');
    var dummy = id('dummyCursor');
    var cx = pageX(dummy) + pageX(ime.elem) - ime.doc.body.scrollLeft;
    var cy = pageY(dummy) + pageY(ime.elem) - ime.doc.body.scrollTop;
    remove(dummy);
    return {x: cx, y: cy};
}

imePopup.commit = function(nextChar)
{
    insertHTML(this.converted);
    this.hideLoadingBox();
    this.hideChoices();

    if(nextChar)
    {
        this.show(nextChar);
    }
    else
    {
        this.hide();
        ime.window.focus();
    }
}

function switchIme()
{
    ime.on = !ime.on;
    if(ime.on)
    {
        id('imeButton').value = 'Turn IME Off';
        id('imeStatus').innerHTML = 'on';
        ime.doc.body.style.background = 'azure';
    }
    else
    {
        id('imeButton').value = 'Turn IME On';
        id('imeStatus').innerHTML = 'off';
        ime.doc.body.style.background = 'white';
        imePopup.hide();
    }

    if(ime.on || !detect.opera) // if we focus on opera, keyboard
    events are responded, but the default action (printing the chars)
    is not done, so it is better not to focus at all
    {
        ime.window.focus();
    }
}

```

```
}
```

### B.4.3 keyboard.js

```
function CharTable(obj)
{
  if(obj)
  {
    for(var code in obj)
    {
      this[code] = obj[code];
    }
  }
};

CharTable.prototype.clone = function()
{
  return new CharTable(this);
};

CharTable.prototype.add = function(code, value)
{
  this[code] = value;
};

CharTable.prototype.remove = function(code)
{
  delete this[code];
};

CharTable.prototype.replace = function(oldCode, newCode)
{
  this[newCode] = this[oldCode];
  delete this[oldCode];
};

var charTable = {
  special: {},
  printable: {}
};

charTable.special.firefox = new CharTable({
  27: 'esc',
  112: 'f1',
  113: 'f2',
  114: 'f3',
  115: 'f4',
  116: 'f5',
  117: 'f6',
  118: 'f7',
  119: 'f8',
  120: 'f9',
  121: 'f10',
  122: 'f11',
  123: 'f12',
```

```

    8: 'backspace',
    9: 'tab',
   13: 'enter',
   20: 'capslock',
   16: 'shift',
   17: 'ctrl',
   91: 'winkey',
   18: 'alt',
   92: 'winkey',
   93: 'propkey',
  145: 'scrolllock',
   19: 'pause',
   45: 'insert',
   36: 'home',
   33: 'pageup',
   46: 'delete',
   35: 'end',
   34: 'pagedown',
   38: 'up',
   37: 'left',
   40: 'down',
   39: 'right',
  144: 'numlock'
});

charTable.special.opera = charTable.special.firefox.clone();

// in opera, 46 (delete) is also ".", 45 (insert) is also "-",
// and 39 (right) is also ""
// prioritize the printable chars because deletion can be
// performed by backspace, insert is not essential,
// and navigation can be done by the mouse
charTable.special.opera.remove(46);
charTable.special.opera.remove(45);
charTable.special.opera.remove(39);

// nonfunctional keys in opera, deleted to prevent conflict with
// printable char keys
charTable.special.opera.remove(18);
charTable.special.opera.remove(20);
charTable.special.opera.remove(91);
charTable.special.opera.remove(92);

// property key
charTable.special.opera.replace(93, 0);

charTable.special.explorer = charTable.special.firefox.clone();

charTable.printable.firefox = new CharTable({
  192: '`~',
  49: '1!',
  50: '2@',
  51: '3#',
  52: '4$',
  53: '5%',
  54: '6^',

```

```

55: '7&',
56: '8*',
57: '9(',
48: '0)',
109: '-_', // TODO: if from numpad, shift shouldn't do anything
61: '=+',
220: '\\|',

219: '[{',
221: ']}',
59: ';:',
222: '\\"',
188: '<',
190: '>',
191: '/?',

//numpad
111: '/',
106: '*',
107: '+',

96: '0',
97: '1',
98: '2',
99: '3',
100: '4',
101: '5',
102: '6',
103: '7',
104: '8',
105: '9',
110: '.'
});

charTable.printable.opera = charTable.printable.firefox.clone();

// `
charTable.printable.opera.replace(192, 96);

// -
charTable.printable.opera.replace(109, 45);
// TODO: if from numpad, shift shouldn't do anything

// \
charTable.printable.opera.replace(220, 92);

// [
charTable.printable.opera.replace(219, 91);

// ]
charTable.printable.opera.replace(221, 93);

// '
charTable.printable.opera.replace(222, 39);

// ,

```

```

charTable.printable.opera.replace(188, 44);

// '
charTable.printable.opera.replace(190, 46);

// /
charTable.printable.opera.replace(191, 47);
// TODO: if from numpad, shift shouldn't do anything

// * (numkey)
charTable.printable.opera.replace(106, 42);

// + (numkey)
charTable.printable.opera.replace(107, 43);

// TODO: for numbers (0-9), if from numpad, shift shouldn't do
anything

charTable.printable.explorer =
charTable.printable.firefox.clone();

// -
charTable.printable.explorer.replace(109, 189);

// +
charTable.printable.explorer.replace(61, 187);

// +
charTable.printable.explorer.replace(59, 186);

```

#### B.4.4 misc.js

```

// The functions here are developed as helper functions to the IME

/*
Description:
  Removes text elements in the DOM with only white spaces
Usage:
  cleanWhiteSpace()
    cleans white space text elements in the whole DOM
  cleanWhiteSpace(elem)
    cleans white space text elements in the provided element
Notes:
  Warning, it will remove intentional white spaces such as the ones in: <p><span>This</span>
<span>is</span> <span>something!</span></p>
*/
function cleanWhiteSpace(elem)
{
  elem = elem || document;
  var child = elem.firstChild;
  while(child != null)
  {
    var next = child.nextSibling;

```

```

        if(child.nodeType == 3 && ! /\S/.test(child.nodeValue)) // 3 is text element, \S is non-
whitespace
        {
            elem.removeChild(child);
        }
        else if(child.nodeType == 1) // 1 is non-text element like p, a, etc.
        {
            cleanWhiteSpace(child);
        }
        child = next;
    }
}
}
/*
Description:
    Returns the first preceding non-text node
*/
function prev(elem)
{
    do
    {
        elem = elem.previousSibling;
    }while(elem && elem.nodeType != 1);
    return elem;
}
/*
Description:
    Returns the first succeeding non-text node
*/
function next(elem)
{
    do
    {
        elem = elem.nextSibling;
    }while(elem && elem.nodeType != 1);
    return elem;
}
/*
Description:
    Returns the first non-text child node
*/
function first(elem)
{
    elem = elem.firstChild;
    return elem && (elem.nodeType == 1 ? elem : next(elem));
}
/*
Description:
    Returns the last non-text child node
*/
function last(elem)

```

```

{
    elem = elem.lastChild;
    return elem && (elem.nodeType == 1 ? elem : prev(elem));
}
/*
Description:
    Returns the n-th parent node, 1 if not specified
Named domParent because parent causes trouble with AJAX on FF
*/
function domParent(elem, num)
{
    num = num || 1;
    while(elem && (num-- > 0))
    {
        elem = elem.parentNode;
    }
    return elem;
}
/*
Description:
    Returns the element with the specified id
    id(ID);
    Returns the element with the specified id inside a specific document
    id(DOCUMENT, ID);
    This function will search down iframes recursively
    (must be delierately programmed, because iframes are considered leaf nodes in the DOM)
*/
function id(doc, name)
{
    var d = (name && doc) || document;
    var n = name || doc;
    var elem = d.getElementById(n);
    if(elem)
    {
        return elem;
    }
    else
    {
        var iframes = d.getElementsByTagName('iframe');
        var tempElem = null;
        for(var i = 0; i < iframes.length; i++)
        {
            tempElem = id(iframes[i].contentWindow.document, n);
            if(tempElem)
            {
                return tempElem;
            }
        }
        return elem;
    }
}
}

```

```

/*
Description:
    Returns the elements with the specified tag name
*/
function tag(name, elem)
{
    return (elem || document).getElementsByName(name);
}

/*
Description:
    Returns the textual content of a node
*/
function text(elem)
{
    var result = "";

    // if elem is not a node, assume it is an array of nodes
    elem = elem.childNodes || elem;

    for(var i = 0; i < elem.length; i++)
    {
        result += elem[i].nodeType == 3 ? elem[i].nodeValue : text(elem[i]);
    }

    return result;
}

/*
Description:
    Checks whether 'elem' has the attribute 'name'
*/
function hasAttribute(elem, name)
{
    return elem.getAttribute(name) != null;
}

/*
Description:
    Gets or set an element's attribute
Usage:
    attr(elem, name)
    attr(elem, name, value)
*/
function attr(elem, name, value)
{
    if(typeof value != "undefined")
    {
        elem.setAttribute(name, value);
    }
    return elem.getAttribute(name);
}

/*

```

Description:

Creates a DOM element, supplying the correct namespace if possible

```

*/
function create(elem)
{
    return document.createElementNS ?
        document.createElementNS("http://www.w3.org/1999/xhtml", elem) :
        document.createElement(elem);
}

```

/\*

Description:

Creates a DOM text element with the supplied content

```

*/
function createText(content)
{
    return document.createTextNode(content);
}

```

/\*

Description:

If a string is provided, will return an array of the string parsed into DOM elements

If a DOM element is provided, will return an array consisting solely of that element

```

*/
function checkElem(arg)
{
    var result = [];
    if(arg.constructor !== Array)
    {
        arg = [arg];
    }
    for(var i = 0; i < arg.length; i++)
    {
        if(arg[i].constructor === String)
        {
            var div = create("div");
            div.innerHTML = arg[i];
            for(var j = 0; j < div.childNodes.length; j++)
            {
                result[result.length] = div.childNodes[j];
            }
        }
        else
        {
            result[result.length] = arg[i];
        }
    }
    return result;
}

```

/\*

Description:

Inserts a DOM element before another one

Usage:

```
before(before, elem)
```

```
before(parent, before, elem)
```

Notes:

'elem' can also be a plain string, which will be parsed as HTML

You can also give an array of DOM elements and strings

Normally 'parent' need not be supplied because the function is intelligent enough to find it

```
*/
function before(parent, before, elem)
{
  if(elem == null)
  {
    elem = before;
    before = parent;
    parent = before.parentNode;
  }

  var elems = checkElem(elem);

  for(var i = 0; i < elems.length; i++)
  {
    parent.insertBefore(elems[i], before);
  }
}

```

/\*

Description:

Adds a DOM element as the last child of another element

Notes:

'elem' can also be a plain string, which will be parsed as HTML

You can also give an array of DOM elements and strings

```
*/
function append(parent, elem)
{
  var elems = checkElem(elem);

  for(var i = 0; i < elems.length; i++)
  {
    parent.appendChild(elems[i]);
  }
}

```

/\*

Description:

Adds a DOM element as the first child of another element

Notes:

'elem' can also be a plain string, which will be parsed as HTML

You can also give an array of DOM elements and strings

```
*/
function prepend(parent, elem)
{
  firstChild = parent.firstChild;
  if(firstChild)
  {

```

```

        before(firstChild, elem);
    }
    else
    {
        append(parent, elem);
    }
}
/*
Description:
    Removes a node from the DOM
*/
function remove(elem)
{
    if(elem)
    {
        elem.parentNode.removeChild(elem);
    }
}
/*
Description:
    Removes all child of a DOM element
*/
function empty(elem)
{
    while(elem.firstChild)
    {
        remove(elem.firstChild);
    }
}
/*
Description:
    Finds the first descendant of parent with the specified id
*/
function descendant(parent, id)
{
    var childs = new Array();
    var current = first(parent);
    while(current)
    {
        if(current.id == id)
        {
            return current;
        }
        childs.push(current);
        current = next(current);
    }
    for(var i = 0; i < childs.length; i++)
    {
        var result = descendant(childs[i], id);
        if(result)
        {

```

```

        return result;
    }
}
return null;
}
/*
START CSS
*/
// gets the current (actual) style
function getStyle(elem, name)
{
    if(elem.style[name])
    {
        return elem.style[name];
    }
    else if(elem.currentStyle) // IE
    {
        return elem.currentStyle[name];
    }
    else if(document.defaultView && document.defaultView.getComputedStyle)
    {
        // g means global, to search all occurrences instead of just one
        name = name.replace(/([A-Z])/g, "-$1").toLowerCase(); // i.e., textAlign becomes text-align
        var s = document.defaultView.getComputedStyle(elem, "");
        return s && s.getPropertyValue(name);
    }
    else
    {
        return null;
    }
}

// gets the x position of an element relative to the page
function pageX(elem)
{
    return elem.offsetParent ?
        elem.offsetLeft + pageX(elem.offsetParent) :
        elem.offsetLeft;
}

// gets the y position of an element relative to the page
function pageY(elem)
{
    return elem.offsetParent ?
        elem.offsetTop + pageY(elem.offsetParent) :
        elem.offsetTop;
}

// gets the x position of an element relative to its DOM parent
function parentX(elem)
{
    return elem.parentNode == elem.offsetParent ?

```

```
    elem.offsetLeft :
    pageX(elem) - pageX(elem.parentNode);
}

// gets the y position of an element relative to its DOM parent
function parentY(elem)
{
    return elem.parentNode == elem.offsetParent ?
        elem.offsetTop :
        pageY(elem) - pageY(elem.parentNode);
}

// gets the x position of an element relative to its CSS container
// e.g., set from left: "100px" in the CSS
// (need not be the DOM parent)
// Note: must be set by CSS, else non-mozilla will fail
function posX(elem)
{
    return parseInt(getStyle(elem, 'left'));
}

// gets the y position of an element relative to its CSS container
// e.g., set from top: "100px" in the CSS
// (need not be the DOM parent)
// Note: must be set by CSS, else non-mozilla will fail
function posY(elem)
{
    return parseInt(getStyle(elem, 'top'));
}

function setX(elem, pos)
{
    elem.style.left = pos + "px";
}

function setY(elem, pos)
{
    elem.style.top = pos + "px";
}

function addX(elem, pos)
{
    setX(elem, posX(elem) + pos);
}

function addY(elem, pos)
{
    setY(elem, posY(elem) + pos);
}

function getWidth(elem)
{
    return parseInt(getStyle(elem, 'width'));
}
```

```
function getHeight(elem)
{
    return parseInt(getStyle(elem, 'height'));
}

function fullHeight(elem)
{
    if(getStyle(elem, 'display') != 'none')
    {
        return elem.offsetHeight || getHeight(elem);
    }

    var old = resetCSS(elem, {
        display: "",
        visibility: 'hidden',
        position: 'absolute'
    });

    var h = elem.clientHeight || getHeight(elem);

    restoreCSS(elem, old);

    return h;
}

function fullWidth(elem)
{
    if(getStyle(elem, 'display') != 'none')
    {
        return elem.offsetWidth || getWidth(elem);
    }

    var old = resetCSS(elem, {
        display: "",
        visibility: 'hidden',
        position: 'absolute'
    });

    var w = elem.clientWidth || getWidth(elem);

    restoreCSS(elem, old);

    return w;
}

function resetCSS(elem, prop)
{
    {
        var old = {};

        for(var i in prop)
        {
            old[i] = elem.style[i];
            elem.style[i] = prop[i];
        }
    }
}
```

```
    return old;
  }

function restoreCSS(elem, prop)
{
  for(var i in prop)
  {
    elem.style[i] = prop[i];
  }
}

function hide(elem)
{
  var curDisplay = getStyle(elem, 'display');

  if(curDisplay != 'none')
  {
    elem.$oldDisplay = curDisplay;
  }
  elem.style.display = 'none';
}

function show(elem)
{
  elem.style.display = elem.$oldDisplay || "";
}

function setOpacity(elem, level)
{
  if(elem.filters)
  {
    elem.style.filter = 'alpha(opacity=' + level + ')';
  }
  else
  {
    elem.style.opacity = level / 100;
  }
}

function debug(msg, replace)
{
  elem = document.getElementById("debug");
  if(elem)
  {
    if(replace == true)
    {
      elem.innerHTML = msg;
    }
    else
    {
      elem.innerHTML += msg + "<br />";
    }
  }
}
```

```
// to handle IE 6 which can't handle the array indexing syntax
String.prototype.at = function(i)
{
  if(i < 0 || i >= this.length)
  {
    return undefined;
  }
  else
  {
    return this.substring(i, i + 1);
  }
}
```

### B.4.5 popup.css

```
.popupNumber{
  border-right: solid rgb(100,100,100) 1px;
  padding: 2px;
  padding-left: 5px;
  padding-right: 5px;
  cursor: pointer;
}

.popupNumberDisabled{
  border-right: solid rgb(100,100,100) 1px;
  padding: 2px;
  padding-left: 5px;
  padding-right: 5px;
  color: gray;
}

.popupChoice{
  padding: 2px;
  padding-right: 20px;
  cursor: pointer;
}

.popupChoiceSelected{
  padding: 2px;
  padding-right: 20px;
  background: darkblue;
  color: white;
  cursor: pointer;
}

.popupChoiceDisabled{
  padding: 2px;
  padding-right: 20px;
}
```

### B.4.6 transliterator.js

```
var romajiToHiragana = "a あ;i い;u う;e え;o お;xa あ;xi い;xu
```

う;xe え;xo お;ka か;ki き;ku く;ke け;ko こ;kka つか;kki つき;kku っ  
 く;kke つけ;kko っこ;kya きや;kyi きい;kyu きゅ;kye きえ;kyo きょ;kkya  
 っきや;kkyi っきい;kkyu っきゅ;kkye っきえ;kkyo っきょ;ga が;gi ぎ;gu  
 ぐ;ge げ;go ご;gga っが;ggi っぎ;ggu っぐ;gge っげ;ggo っご;gya  
 ぎや;gyi ぎい;gyu ぎゅ;gye ぎえ;gyo ぎょ;ggya っぎや;ggyi っぎい;ggyu っ  
 ぎゅ;ggye っぎえ;ggyo っぎょ;sa さ;si し;su す;se せ;so そ;ssa っ  
 さ;ssi っし;ssu っす;sse っせ;sso っそ;sya しや;syi しい;syu しゅ;sye  
 しゃ;syo しょ;ssya っしや;ssyi っしい;ssyu っしゅ;ssye っしえ;ssyo っ  
 しよ;sha しゃ;shi し;shu しゅ;she しゃ;sho しょ;ssha っしや;sshi っ  
 し;sshu っしゅ;sshe っしえ;ssho っしよ;za ざ;zi じ;zu ず;ze ぜ;zo  
 ぞ;zza っざ;zzi っじ;zzu っず;zze っぜ;zso っぞ;zya じゃ;zui じい;zju  
 じゅ;zye じゃ;zyo じょ;zzya っじゃ;zzyi っじい;zzyu っじゅ;zzye っ  
 じえ;zzyo っじよ;ja じゃ;ji じ;ju じゅ;je じゃ;jyo じょ;jja っじゃ;jji っ  
 じ;jju っじゅ;jje っじえ;jjo っじょ;jya じゃ;jyi じい;jyu じゅ;jye じゃ;  
 jyo じょ;jjya っじゃ;jjyi っじい;jjyu っじゅ;jjye っじえ;jjyo っじょ;ta  
 た;ti ち;tu っ;te て;to と;tta った;tti っち;ttu っつ;tte って;tto っ  
 と;xtu っ;tha てや;thi てい;thu てゅ;the てえ;tho てよ;ttha っ  
 てや;tthi ってい;tthu ってゅ;tthe ってえ;ttho ってよ;tya ちゃ;tyi  
 ちい;tyu ちゅ;tye ちえ;tyo ちよ;ttya っちや;ttyi っちい;ttyu っ  
 ちゅ;ttye っちえ;ttyo っちよ;cha ちゃ;chi ち;chu ちゅ;che ちえ;cho ちよ;  
 ccha っちや;cchi っちい;cchu っちゅ;cche っちえ;ccho っちよ;cya ちゃ;cyi  
 ちい;cyu ちゅ;cye ちえ;cyo ちよ;ccya っちや;ccyi っちい;ccyu っ  
 ちゅ;ccye っちえ;ccyo っちよ;tsu っ;ttsu っつ;da だ;di ぢ;du づ;de  
 で;do だ;dda っだ;ddi っぢ;ddu っづ;dde っで;ddo っだ;dya ぢや;dyl  
 ぢい;dyl ぢゅ;dye ぢえ;dyl ぢよ;ddy っぢや;ddyl っぢい;ddyl っ  
 ぢゅ;ddye っぢえ;ddyl ぢよ;dha ぢや;dhi ぢい;dhu ぢゅ;dhe ぢえ;dho  
 ぢよ;ddha っぢや;ddhi っぢい;ddhu っぢゅ;ddhe っぢえ;ddho っぢよ;na  
 な;ni に;nu ぬ;ne ね;no の;nya にや;nyi にい;nyu にゅ;nye にえ;nyo  
 によ;ha は;hi ひ;hu ふ;he へ;ho ほ;hha っは;hhi っひ;hhu っふ;hhe っへ;  
 hho っほ;hya ひや;hyi ひい;hyu ひゅ;hye ひえ;hyo ひよ;hhya っひや;hhyi  
 っひい;hhyu っひゅ;hhye っひえ;hhyo っひよ;fa っふあ;fi っふい;fu っふ;  
 fe っふえ;fo っふお;ffa っふあ;ffi っふい;ffu っふふ;ffe っふえ;ffo っふお;fy  
 っふや;fyi っふい;fyu っふゅ;fye っふえ;fyo っふよ;ffya っふや;ffyi っふい;ffyu っ  
 ふゅ;ffye っふえ;ffyo っふよ;ba っば;bi っび;bu っぶ;be っべ;bo っぼ;bb  
 っば;bbi っび;bbu っぶ;bbe っべ;bbo っぼ;bya っびや;byi っびい;byu っびゅ;bye  
 っびえ;byo っびよ;bbya っびや;bbyi っびい;bbyu っびゅ;bbye っびえ;bbyo っ  
 びよ;pa っぱ;pi っぴ;pu っぷ;pe っぺ;po っぽ;ppa っぱ;ppi っぴ;ppu っぷ;ppe っ  
 っぺ;ppo っっぱ;pya っぴや;pyi っぴい;pyu っぴゅ;pye っぴえ;pyo っぴよ;ppy  
 っぴや;ppyi っぴい;ppyu っぴゅ;ppy っぴえ;ppy っぴよ;ma っま;mi っみ;me っめ;mo  
 っも;mma っま;mml っみ;mmu っむ;mme っめ;mml っも;mya っみや;myi っみい;myu  
 っみゅ;mye っみえ;myo っみよ;mmya っみや;mmyi っみい;mmyu っみゅ;mmye っ  
 みえ;mmyo っみよ;ya っや;yi っい;yu っゅ;ye っえ;yoy っよ;yya っや;yyl  
 っい;yyl っゅ;yyl っえ;yyl っよ;xya っや;xyu っゅ;xyo っよ;ra っら;ri っり;ru っ  
 る;re っれ;ro っろ;rra っら;rri っり;rru っる;rre っれ;rro っろ;rya っりや;ryi  
 っりい;ryu っりゅ;rye っりえ;ryo っりよ;rrya っりや;rryi っりい;rryu っ  
 りゅ;rrye っりえ;rryo っりよ;wa っわ;wi っうい;wu っう;we っうえ;wo っを;xwa  
 っわ;wwa っわ;wwi っうい;wwu っう;wwe っうえ;wwo っを;wha っわ;whi  
 っうい;whu っう;whe っうえ;who っうお;wwha っわ;wwhi っうい;wwhu っう;wwhe っ  
 うえ;wwho っうお;va っゎ;vi っゎい;vu っゎ;ve っゎえ;vo っゎお;vva っゎあ;vvi っ  
 ゎい;vvu っゎ;vve っゎえ;vvo っゎお;vya っゎや;vyi っゎい;vyu っゎゅ;vye っゎえ;  
 vyo っゎよ;vvy っゎや;vvyi っゎい;vvyu っゎゅ;vvye っゎえ;vvyo っゎよ;n  
 ん;n' ん;nn ん";

var romajiToKatakana = "a ア;i イ;u ウ;e エ;o オ;xa ア;xi イ;xu  
 う;xe エ;xo オ;ka カ;ki キ;ku ク;ke ケ;ko コ;kka ツカ;kki ツキ;kku ツ  
 く;kke ツケ;kko ツコ;kya キヤ;kyi キイ;kyu キユ;kye キエ;kyo キヨ;kkya

ッキヤ;kkyi ッキイ;kkyu ッキユ;kkye ッキエ;kkyo ッキヨ;ga ガ;gi ギ;gu  
 グ;ge ゲ;go ゴ;gga ッガ;ggi ッギ;ggg ッグ;gge ッゲ;ggo ッゴ;gya  
 ギヤ;gyi ギイ;gyu ギユ;gye ギエ;gyo ギヨ;ggya ッギヤ;ggyi ッギイ;ggyu ッ  
 ギユ;ggye ッギエ;ggyo ッギヨ;sa サ;si シ;su ス;se セ;so ソ;ssa ッ  
 サ;ssi ッシ;ssu ッス;sse ッセ;sso ッソ;sya シヤ;syi シイ;syu シユ;syu  
 シエ;syo シヨ;ssya ッシヤ;ssyi ッシイ;ssyu ッシユ;ssye ッシエ;ssyo ッ  
 シヨ;sha シャ;shi シ;shu シュ;she シエ;sho シヨ;ssha ッシャ;sshi ッ  
 シ;sshu ッシユ;sshe ッシエ;ssho ッシヨ;za ザ;zi ジ;zu ズ;ze ゼ;zo  
 ゴ;zza ッザ;zzi ッジ;zzu ッズ;zze ッゼ;zzo ッゾ;zya ジャ;zyi ジイ;zyu  
 ジュ;zye ジエ;zyo ジヨ;zzya ッジャ;zzyi ッジイ;zzyu ッジュ;zzye ッ  
 ジエ;zzyo ッジヨ;ja ジャ;ji ジ;ju ジュ;je ジエ;jo ジヨ;jja ッジャ;jji ッ  
 ジ;jju ッジュ;jje ッジエ;jjo ッジヨ;jya ジャ;jyi ジイ;jyu ジュ;jye ジエ;  
 jyo ジヨ;jjya ッジャ;jjyi ッジイ;jjyu ッジュ;jjye ッジエ;jjyo ッジヨ;ta  
 タ;ti チ;tu ツ;te テ;to ト;tta ッタ;tti ッチ;ttu ッツ;tte ッテ;tto ッ  
 ト;xtu ッ;tha テヤ;thi テイ;thu テユ;the テエ;tho テヨ;ttha ッ  
 テヤ;tthi ッテイ;tthu ッテユ;tthe ッテエ;ttho ッテヨ;tya チャ;tyi  
 チイ;tyu チユ;tye チエ;tyo チヨ;ttya ッチャ;ttyi ッチイ;ttyu ッ  
 チユ;ttye ッチエ;ttyo ッチヨ;cha チャ;chi チ;chu チュ;che チエ;cho チヨ;  
 ccha ッチャ;cchi ッチ;cchu ッチュ;cche ッチエ;ccho ッチヨ;cya チャ;cyi  
 チイ;cyu チユ;cye チエ;cyo チヨ;ccya ッチャ;ccyi ッチイ;ccyu ッ  
 チユ;ccye ッチエ;ccyo ッチヨ;tsu ツ;ttu ッツ;da ダ;di チ;du ッ;  
 de デ;do ド;dda ッダ;ddi ッチ;ddu ッヅ;dde ッデ;ddo ッド;dya チャ;dyi  
 チイ;dyu チユ;dye チエ;dyo チヨ;ddya ッチャ;ddyi ッチイ;ddyu ッ  
 チユ;ddye ッチエ;ddyo ッチヨ;dha デヤ;dhi デイ;dhu デユ;dhe デエ;dho  
 デヨ;ddha ッデヤ;ddhi ッデイ;ddhu ッデユ;ddhe ッデエ;ddho ッデヨ;na  
 ナ;ni ニ;nu ヌ;ne ネ;no ノ;nya ニヤ;nyi ニイ;nyu ニユ;nye ニエ;nyo  
 ニヨ;ha ハ;hi ヒ;hu フ;he ヘ;ho ホ;hha ッハ;hhi ッヒ;hhu ッフ;hhe ッヘ;  
 hho ッホ;hya ヒヤ;hyi ヒイ;hyu ヒユ;hye ヒエ;hyo ヒヨ;hhya ッヒヤ;hhyi  
 ッヒイ;hhyu ッヒユ;hhye ッヒエ;hhyo ッヒヨ;fa ファ;fi フィ;fu フ;fe  
 フェ;fo フォ;ffa ッファ;ffi ッフィ;ffu ッフ;ffe ッフェ;ffo ッフォ;fya  
 ファ;fyi フィ;fyu フユ;fye フェ;fyo フヨ;ffya ッファ;ffyi ッフィ;ffyu ッ  
 フユ;ffye ッフェ;ffyo ッフォ;ba バ;bi ビ;bu ブ;be ベ;bo ボ;bba ッ  
 バ;bbi ッビ;bbu ッブ;bbe ッベ;bbo ッボ;bya ビヤ;byi ビイ;byu ビユ;bye  
 ビエ;byo ビヨ;bbya ッビヤ;bbyi ッビイ;bbyu ッビユ;bbye ッビエ;bbyo ッ  
 ビヨ;pa パ;pi ピ;pu プ;pe ペ;po ポ;ppa ッパ;ppi ッピ;ppu ップ;ppe ッペ;  
 ppo ッポ;pya ピヤ;pyi ピイ;pyu ピユ;pye ピエ;pyo ピヨ;ppya ッピヤ;ppyi  
 ッピイ;ppyu ッピユ;ppye ッピエ;ppyu ッピヨ;ma マ;mi ミ;mu ム;me メ;mo  
 モ;mma ッマ;mmi ッミ;mmu ッム;mme ッメ;mmo ッモ;mya ミヤ;myi ミイ;myu  
 ミユ;mmye ッミエ;mmyo ッミヨ;mmya ッミヤ;mmyi ッミイ;mmyu ッミユ;mmye ッ  
 ミエ;mmyo ッミヨ;ya ヤ;yi イ;yu ヌ;ye イエ;yya ッヤ;yyi ッイ;yyu  
 ッユ;yye ッイエ;yyo ッヨ;xya ヤ;xyu ヌ;xyo ヌ;ra ラ;ri リ;ru ル;re  
 レ;ro ロ;rra ッラ;rrr ッリ;rru ッル;rre ッレ;rro ッロ;rya リヤ;ryi  
 リイ;ryu リユ;rye リエ;ryo リヨ;rrya ッリヤ;rryi ッリイ;rryu ッ  
 リユ;rrye ッリエ;rryo ッリヨ;wa ワ;wi ウイ;wu ウ;we ウエ;wo ヲ;xwa  
 ワ;wwa ッワ;wwi ッウイ;wwu ッウ;wwe ッウエ;wwo ッワ;wha ウア;whi  
 ウイ;whu ウ;whe ウエ;who ウオ;wwha ッウア;wwhi ッウイ;wwhu ッウ;wwhe ッ  
 ウエ;wwho ッウオ;va ヲ;vi ヲ;vu ヲ;ve ヲ;vo ヲ;vva ッヱ;vvi ッ  
 ヲ;vvy ッヱ;vve ッヱ;vvo ッヱ;vya ヲ;vyi ヲ;vyu ヲ;vye ヲ;vvo ッヱ;vvyo ッヱ;vvyo ッヱ;n  
 ン;n' ン;nn ン;xka カ;xke ケ";

```

var romajiToHalfwidth = "a ア;i イ;u ウ;e エ;o オ;xa ア;xi イ;xu ウ;xe エ;xo  

オ;ka カ;ki キ;ku ク;ke ケ;ko コ;kka ッカ;kki ッキ;kku ック;kke ッケ;kko ッコ;kya カヤ;  

kyl キイ;kyl キユ;kyl キエ;kyl キヨ;kkya ッカヤ;kkyi ッキイ;kkyu ックユ;kkye ッケエ;kkyo  

ッキヨ;ga ガ;gi ギ;gu グ;ge ゲ;go ゴ;gga ッガ;ggi ッギ;ggg ッグ;gge ッゲ;ggo ッゴ;gya  

ッギヤ;gyi ギイ;gyu ギユ;gye ギエ;gyo ギヨ;ggya ッギヤ;ggyi ッギイ;ggyu ッ  


```

```

k`u;ggye ʔk`ɛ;ggyo ʔk`ɔ;sa ʃ;si ʃ;su ʃ;se ʃ;so ʃ;ssa ʃʃ;ssi ʃʃ;ssu ʃʃ;
sse ʃʃ;ssu ʃʃ;syɑ ʃʃ;syi ʃʃ;syu ʃʃ;syɛ ʃʃ;syo ʃʃ;ssya ʃʃʃ;ssyi ʃʃʃ;
ʃʃʃ;ssyu ʃʃʃ;ssye ʃʃʃ;ssyo ʃʃʃ;sha ʃʃʃ;shi ʃʃʃ;shu ʃʃʃ;she ʃʃʃ;sho ʃʃʃ;ssha
ʃʃʃ;sshi ʃʃʃ;sshu ʃʃʃ;sshe ʃʃʃ;ssho ʃʃʃ;za ʃʃʃ;zi ʃʃʃ;zu ʃʃʃ;ze ʃʃʃ;zo
ʃʃʃ;zza ʃʃʃ;zzi ʃʃʃ;zzu ʃʃʃ;zze ʃʃʃ;zzo ʃʃʃ;zyɑ ʃʃʃ;zyi ʃʃʃ;zyu
ʃʃʃ;zyɛ ʃʃʃ;zyo ʃʃʃ;zzya ʃʃʃ;zzyi ʃʃʃ;zzyu ʃʃʃ;zzye ʃʃʃ;zzyo ʃʃʃ;
ʃʃʃ;ja ʃʃʃ;ji ʃʃʃ;ju ʃʃʃ;je ʃʃʃ;jo ʃʃʃ;jja ʃʃʃ;jji ʃʃʃ;jju ʃʃʃ;jje ʃʃʃ;
ʃʃʃ;jjo ʃʃʃ;jya ʃʃʃ;jyi ʃʃʃ;jyu ʃʃʃ;jye ʃʃʃ;jyo ʃʃʃ;jjya ʃʃʃ;jjyi ʃʃʃ;
ʃʃʃ;jjyu ʃʃʃ;jjye ʃʃʃ;jjyo ʃʃʃ;ta ʃʃʃ;ti ʃʃʃ;tu ʃʃʃ;te ʃʃʃ;to ʃʃʃ;tta ʃʃʃ;tti
ʃʃʃ;ttu ʃʃʃ;tte ʃʃʃ;tto ʃʃʃ;xtu ʃʃʃ;tha ʃʃʃ;thi ʃʃʃ;thu ʃʃʃ;the ʃʃʃ;tho
ʃʃʃ;ttha ʃʃʃ;tthi ʃʃʃ;tthu ʃʃʃ;tthe ʃʃʃ;ttho ʃʃʃ;tyɑ ʃʃʃ;tyi ʃʃʃ;tyu
ʃʃʃ;tyɛ ʃʃʃ;tyo ʃʃʃ;ttyɑ ʃʃʃ;ttyi ʃʃʃ;ttyu ʃʃʃ;ttye ʃʃʃ;ttyo ʃʃʃ;cha
ʃʃʃ;chɑ ʃʃʃ;chi ʃʃʃ;chu ʃʃʃ;che ʃʃʃ;cho ʃʃʃ;cchɑ ʃʃʃ;cchi ʃʃʃ;cchu ʃʃʃ;cche ʃʃʃ;
ʃʃʃ;ccho ʃʃʃ;cya ʃʃʃ;cyi ʃʃʃ;cyu ʃʃʃ;cyɛ ʃʃʃ;cyo ʃʃʃ;ccya ʃʃʃ;ccyi ʃʃʃ;ccyu
ʃʃʃ;ccyɛ ʃʃʃ;ccyo ʃʃʃ;tsu ʃʃʃ;ttsu ʃʃʃ;da ʃʃʃ;di ʃʃʃ;du ʃʃʃ;de ʃʃʃ;do ʃʃʃ;dda
ʃʃʃ;ddi ʃʃʃ;ddu ʃʃʃ;dde ʃʃʃ;ddo ʃʃʃ;dya ʃʃʃ;dyi ʃʃʃ;dyu ʃʃʃ;dye
ʃʃʃ;dyo ʃʃʃ;ddyɑ ʃʃʃ;ddyi ʃʃʃ;ddyu ʃʃʃ;ddye ʃʃʃ;ddyo ʃʃʃ;dha
ʃʃʃ;dhi ʃʃʃ;dhu ʃʃʃ;dhe ʃʃʃ;dho ʃʃʃ;ddha ʃʃʃ;ddhi ʃʃʃ;ddhu ʃʃʃ;ddhe
ʃʃʃ;ddho ʃʃʃ;na ʃʃʃ;ni ʃʃʃ;nu ʃʃʃ;ne ʃʃʃ;no ʃʃʃ;nya ʃʃʃ;nyi ʃʃʃ;nyu ʃʃʃ;nye
ʃʃʃ;nyo ʃʃʃ;ha ʃʃʃ;hi ʃʃʃ;hu ʃʃʃ;he ʃʃʃ;ho ʃʃʃ;hha ʃʃʃ;hhi ʃʃʃ;hhu ʃʃʃ;hhe ʃʃʃ;hho ʃʃʃ;
ʃʃʃ;hya ʃʃʃ;hyi ʃʃʃ;hyu ʃʃʃ;hyɛ ʃʃʃ;hyo ʃʃʃ;hha ʃʃʃ;hhyi ʃʃʃ;hhyu ʃʃʃ;hhye
ʃʃʃ;hhyo ʃʃʃ;fa ʃʃʃ;fi ʃʃʃ;fu ʃʃʃ;fo ʃʃʃ;ffa ʃʃʃ;ffi ʃʃʃ;ffu ʃʃʃ;ffe ʃʃʃ;
ʃʃʃ;ffo ʃʃʃ;fya ʃʃʃ;fyi ʃʃʃ;fyu ʃʃʃ;fyɛ ʃʃʃ;fyo ʃʃʃ;ffya ʃʃʃ;ffyi ʃʃʃ;ffyu
ʃʃʃ;ffye ʃʃʃ;ffyo ʃʃʃ;ba ʃʃʃ;bi ʃʃʃ;bu ʃʃʃ;be ʃʃʃ;bo ʃʃʃ;bba ʃʃʃ;bbi ʃʃʃ;
ʃʃʃ;bbu ʃʃʃ;bbe ʃʃʃ;bbo ʃʃʃ;bbyɑ ʃʃʃ;bbyi ʃʃʃ;bbyu ʃʃʃ;bbye ʃʃʃ;bbyo ʃʃʃ;pa ʃʃʃ;pi ʃʃʃ;pu
ʃʃʃ;pe ʃʃʃ;po ʃʃʃ;ppɑ ʃʃʃ;ppi ʃʃʃ;ppu ʃʃʃ;ppe ʃʃʃ;ppo ʃʃʃ;pya ʃʃʃ;pyi ʃʃʃ;pyu
ʃʃʃ;pyɛ ʃʃʃ;pyo ʃʃʃ;ppya ʃʃʃ;ppyi ʃʃʃ;ppyu ʃʃʃ;ppye ʃʃʃ;ppyo ʃʃʃ;
ʃʃʃ;ma ʃʃʃ;mi ʃʃʃ;mu ʃʃʃ;me ʃʃʃ;mo ʃʃʃ;mma ʃʃʃ;mmi ʃʃʃ;mmu ʃʃʃ;mme ʃʃʃ;mmo ʃʃʃ;myɑ
ʃʃʃ;myi ʃʃʃ;myu ʃʃʃ;myɛ ʃʃʃ;myo ʃʃʃ;mmyɑ ʃʃʃ;mmyi ʃʃʃ;mmyu ʃʃʃ;mmye ʃʃʃ;
ʃʃʃ;mmyo ʃʃʃ;ya ʃʃʃ;yɑ ʃʃʃ;yu ʃʃʃ;yɛ ʃʃʃ;yɔ ʃʃʃ;yya ʃʃʃ;yyi ʃʃʃ;yyu ʃʃʃ;yye ʃʃʃ;
ʃʃʃ;yyo ʃʃʃ;xya ʃʃʃ;xyu ʃʃʃ;xyo ʃʃʃ;ra ʃʃʃ;ri ʃʃʃ;ru ʃʃʃ;re ʃʃʃ;ro ʃʃʃ;rrɑ ʃʃʃ;rri ʃʃʃ;
ʃʃʃ;rru ʃʃʃ;rre ʃʃʃ;rro ʃʃʃ;ryɑ ʃʃʃ;ryi ʃʃʃ;ryu ʃʃʃ;ryɛ ʃʃʃ;ryo ʃʃʃ;rrya ʃʃʃ;
ʃʃʃ;rryi ʃʃʃ;rryu ʃʃʃ;rrye ʃʃʃ;rryo ʃʃʃ;wa ʃʃʃ;wi ʃʃʃ;wu ʃʃʃ;we ʃʃʃ;wo ʃʃʃ;xwɑ
ʃʃʃ;wwɑ ʃʃʃ;wwi ʃʃʃ;wwu ʃʃʃ;wwe ʃʃʃ;wwo ʃʃʃ;whɑ ʃʃʃ;whi ʃʃʃ;whu ʃʃʃ;whe
ʃʃʃ;who ʃʃʃ;wwhɑ ʃʃʃ;wwhi ʃʃʃ;wwhu ʃʃʃ;wwhe ʃʃʃ;wwho ʃʃʃ;va ʃʃʃ;vi
ʃʃʃ;vu ʃʃʃ;ve ʃʃʃ;vo ʃʃʃ;vva ʃʃʃ;vvi ʃʃʃ;vvu ʃʃʃ;vve ʃʃʃ;vvo ʃʃʃ;
ʃʃʃ;vya ʃʃʃ;vyi ʃʃʃ;vyu ʃʃʃ;vye ʃʃʃ;vyo ʃʃʃ;vvyɑ ʃʃʃ;vvyi ʃʃʃ;vvyu ʃʃʃ;
ʃʃʃ;vvyɛ ʃʃʃ;vvyo ʃʃʃ;n ʃʃʃ;n' ʃʃʃ;nn ʃʃʃ;- ʃʃʃ;/ ʃʃʃ;";

```

```

var numberToFullwidth = "1 1;2 2;3 3;4 4;5 5;6 6;7 7;8 8;9
9;0 0";

```

```

var capsToFullwidth = "A A;B B;C C;D D;E E;F F;G G;H H;I
I;J J;K K;L L;M M;N N;O O;P P;Q Q;R R;S S;T T;U U;V V;W
W;X X;Y Y;Z Z";

```

```

var smallToFullwidth = "a a;b b;c c;d d;e e;f f;g g;h h;i i;
j j;k k;l l;m m;n n;o o;p p;q q;r r;s s;t t;u u;v v;w
w;x x;y y;z z";

```

```

var symbolToFullwidth = "` `;~ ~;! !;@ @;# #;$ $;% %;^ ^;&
&* *;( ();- -;_ _;= =;+ +;\ \;| |;[ [;{ {;] ];};? ?;
; ;;: :;: ' ;\" \" ;, , ;< <;. . ;> >;/ / ;?? ?;? ?";

```

```
// Transliteration node represents a node in the transliteration
```

```

tree
function TransNode(source, dest)
{
  if(arguments.length == 1)
  {
    dest = null;
  }
  this.source = source;
  this.dest = dest;
  this.next = new Array();
}

function Transliterator(rule)
{
  this.root = new Array();

  if(arguments.length == 1)
  {
    this.addRule(rule);
  }
}

Transliterator.prototype.addRule = function(source, dest)
{
  if(arguments.length == 1)
  {
    return this.addString(source);
  }

  var current = this.root;
  for(var i = 0; i < source.length; i++)
  {
    var found = null;
    var s = source.at(i);

    found = Transliterator.findChild(current, s);

    if(found == null)
    {
      found = new TransNode(s);
      current.push(found);
    }

    current = found.next;
  }
  if(found)
  {
    found.dest = dest;
  }
};

// example input: src1 dest1;src2 dest2;a あ
// escape:
// ?? -> ?
// ?; -> ;
// ?<space> -> <space>

```

```

// ?<other> -> <other>
// actually:
// ?<any> -> <any>
Transliterator.prototype.addString = function(str)
{
    var ruleSrc = new Array();
    var ruleDest = new Array();
    var currentRule = ruleSrc;
    var escapeMode = false;
    for(var i = 0; i < str.length; i++)
    {
        if(escapeMode)
        {
            currentRule.push(str.at(i));
            escapeMode = false;
        }
        else if(str.at(i) == ';')
        {
            // delimiter detected
            this.addRule(ruleSrc.join(''), ruleDest.join(''));
            ruleSrc = new Array();
            ruleDest = new Array();
            currentRule = ruleSrc;
        }
        else if(str.at(i) == '?')
        {
            escapeMode = true;
        }
        else if(str.at(i) == ' ' && currentRule == ruleSrc)
        {
            currentRule = ruleDest;
        }
        else
        {
            currentRule.push(str.at(i));
        }
    }

    this.addRule(ruleSrc.join(''), ruleDest.join(''));
};

Transliterator.prototype.transliterate = function(source)
{
    var dest = new Array(); // better than doing many string
    concats
    var i = 0;

    while (i < source.length)
    {
        var nextNodes = this.root;
        var found = null;
        var lastNodeFound = null;
        var searchStartIndex = i;

        while (i < source.length &&
            (found = Transliterator.findChild(nextNodes,

```

```
source.at(i))) != null)
    {
        lastNodeFound = found;
        nextNodes = lastNodeFound.next;
        i++;
    }

    if(lastNodeFound == null || lastNodeFound.dest == null)
    {
        dest.push(source.at(searchStartIndex));
        i = searchStartIndex + 1;
    }
    else
    {
        dest.push(lastNodeFound.dest);
    }
}

return dest.join('');
};

Transliterator.findChild = function(array, source)
{
    for(var i = 0; i < array.length; i++)
    {
        if(array[i].source == source)
        {
            return array[i];
        }
    }
    return null;
};

var hiraganaTrans = new Transliterator(romajiToHiragana);
hiraganaTrans.addRule(symbolToFullwidth);
hiraganaTrans.addRule(numberToFullwidth);

var katakanaTrans = new Transliterator(romajiToKatakana);
katakanaTrans.addRule(symbolToFullwidth);
katakanaTrans.addRule(numberToFullwidth);

var halfwidthTrans = new Transliterator(romajiToHalfwidth);

var fullwidthTrans = new Transliterator(numberToFullwidth);
fullwidthTrans.addRule(symbolToFullwidth);
fullwidthTrans.addRule(smallToFullwidth);
fullwidthTrans.addRule(capsToFullwidth);
```

## Daftar Isi

LAMPIRAN A - Diagram Kelas.....	3
LAMPIRAN B - Source Code.....	4
B.1 EdictDbBuilder.exe.....	4
B.1.1 Edict.cs.....	4
B.1.2 BuildDb.cs.....	7
B.1.3 edict_allowed_text.txt.....	11
B.1.4 edict_tags_transform.txt.....	12
B.1.5 edict_popularity_tags.txt.....	12
B.2 GamaIme.dll.....	12
B.2.1 Helper.cs.....	13
B.2.2 HiraganaToRomajiTransliterator.cs.....	13
B.2.3 KanjiTransliterator.cs.....	14
B.2.4 RestrictionChecker.cs.....	23
B.2.5 RomajiToKanaTransliterator.cs.....	24
B.2.6 Rule.cs.....	25
B.2.7 RuleInstance.cs.....	28
B.2.8 StringTransliterator.cs.....	30
B.2.9 Transliterator.cs.....	31
B.2.10 edict_tags.txt.....	37
B.2.11 hiragana_to_romaji.txt.....	38
B.2.12 restriction.txt.....	41
B.2.13 romaji_to_hiragana.txt.....	42
B.2.14 romaji_to_katakana.txt.....	50
B.2.15 rules_romaji.txt.....	59
B.3 transliterator.aspx.....	64
B.4 Aplikasi Web IME.....	64
B.4.1 ime.htm.....	64
B.4.2 ime.js.....	65
B.4.3 keyboard.js.....	80
B.4.4 misc.js.....	83
B.4.5 popup.css.....	94
B.4.6 transliterator.js.....	94