ダジック・アースコンテンツ作成補助ツール

Demic/Desicの紹介と利用法

岩城邦典、津川卓也（国立研究開発法人情報通信研究機構）

ダジック・アースファイル一式

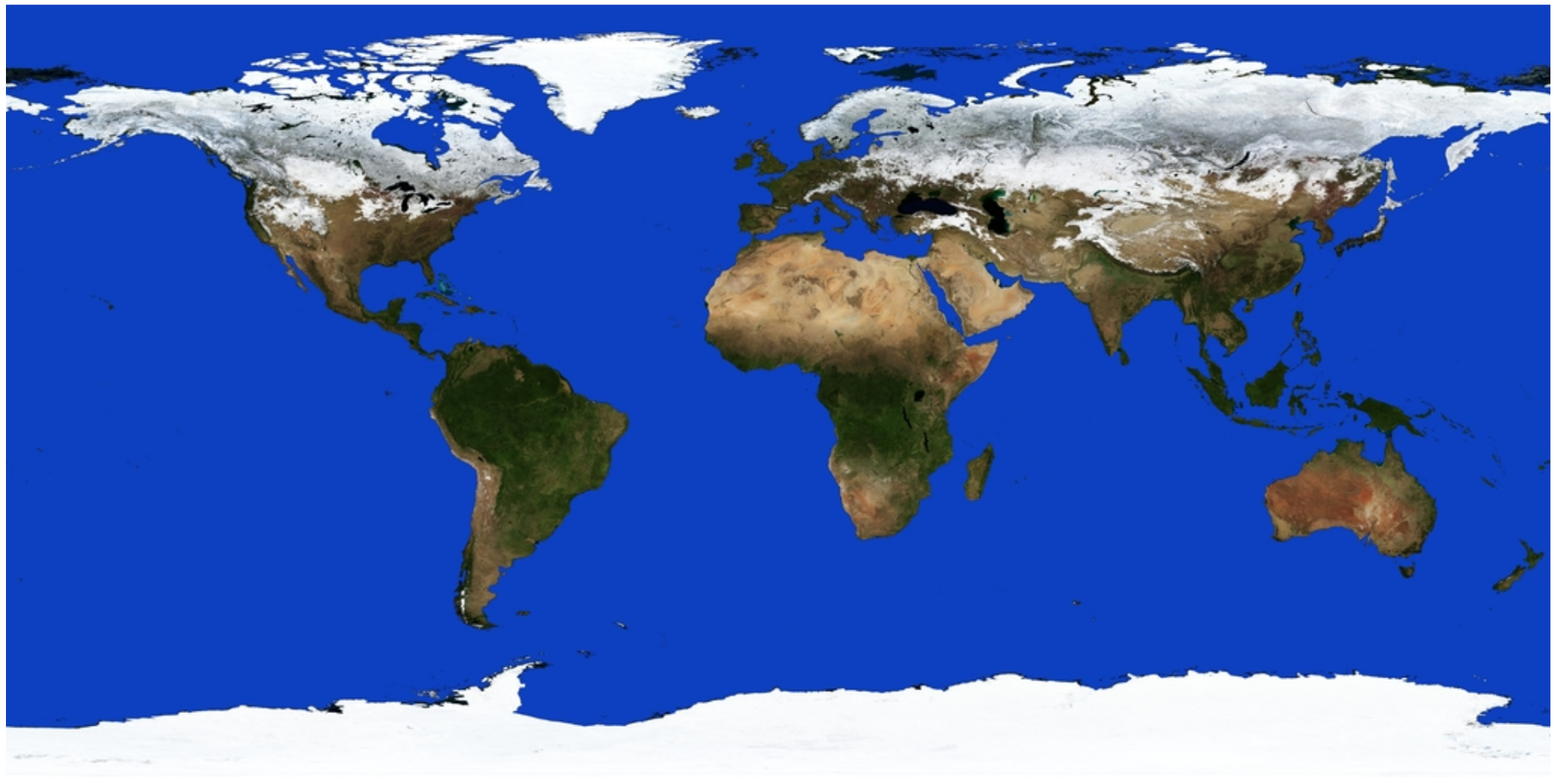
- 配布DVDなどに含まれるDagik_Earth_folder/misc/Dagik_template（基本コンテンツ構成になっています）をコピーするか、ダジックアースWEBのダウンロードページからコンテンツ作成用サンプルをダウンロード（http://dagik.org/DE/menu/misc/Dagik_template.zip）して、ダジック・アースファイル一式を用意します。

```
Dagikフォルダ/
    ├Dagik_Earth.exe（Windows版実行ファイル：64bitで動作させる場合はDagik_Earth_64bit.exeを実行します）
    ├Dagik_Earth.app（Mac版実行ファイル：Mac_Dagik_Earth_app.zipを解凍します）
    ├index.html（WEB版実行ファイル）
    └data/
        ├js/（WEB版用javascript）
        ├draw/（ライン・矢印等描画ファイル）
        ├plot/（お絵かきデータ保存フォルダ）
        ├icons/（UI用アイコン）
        ├conf/（設定ファイル）
        │    ├lang-xxxx.txt（言語ごとの設定）
        │    ├init_conf.txt（投影環境（ハードウェアなど）ごとの設定）
        │    ├dow_conf.txt（WEB版コンテンツ設定）
        │    └conf.txt（Windows/Mac版コンテンツ設定）
        └images/（イメージファイル）
            ├title.gif（メニュー画面用タイトル画像）
            ├map/（地球画像フォルダ）
            │    ├map_0.jpg
            │    ├  ⋮
            │    └map_***.jpg
            └screen/（キャプション画像フォルダ）
                 ├screen_0.jpg
                 ├  ⋮
                 └screen_***.jpg
```

ダジック・アースファイル一式の構成内容

ダジック・アースコンテンツ制作に必要なもの

- オリジナルのダジックアースコンテンツを作るにあたって、正距円筒図法で描かれた地球画像とキャプション画像が必要です。（2019年版ダジック・アースマニュアルp.71参照）



地球画像
(正距円筒図法:*2048×1024px*)

キャプション画像
(*512×1024px*)

地球画像の設定

- 作成した正距円筒図法の画像をmap_0.jpgというファイルネームで（動画であるならば、各コマの正距円筒図法にmap_1.jpg、map_2.jpgなどの連番ファイルネームで）、Dagikフォルダのdata/images/map/に保存します。

```
Dagikフォルダ/
　└data/
　　└images/（イメージファイル）
　　├map/（地球画像フォルダ）
　　|　　├map_0.jpg
　　|　　├　⋮
　　|　　└map_***.jpg
　　└screen/（キャプション画像フォルダ）
　　　　├screen_0.jpg
　　　　├　⋮
　　　　└screen_***.jpg
```

←ここに地球画像を入れます。

キャプション画像の設定

- 作成したキャプション画像をscreen_0.jpgというファイルネームで（動画であるならば、各コマのキャプション画像にscreen_1.jpg、screen_2.jpgなどの連番ファイルネームで）、Dagikフォルダのdata/images/screen/に保存します。

```
Dagikフォルダ/
　└data/
　　└images/（イメージファイル）
　　├map/（地球画像フォルダ）
　　│　　├map_0.jpg
　　│　　├　：
　　│　　└map_***.jpg
　　└screen/（キャプション画像フォルダ）
　　　　├screen_0.jpg
　　　　├　：
　　　　└screen_***.jpg
```

←ここにキャプション画像を入れます。

設定ファイルと実行

- Dagikフォルダのdata/conf/に各種設定ファイルがあります。
  必要であれば、設定ファイルの値を書き換えます。
  （詳細は2019年版ダジック・アースマニュアルp.77参照）

- それぞれの環境に合わせって、実行ファイルを実行させます。

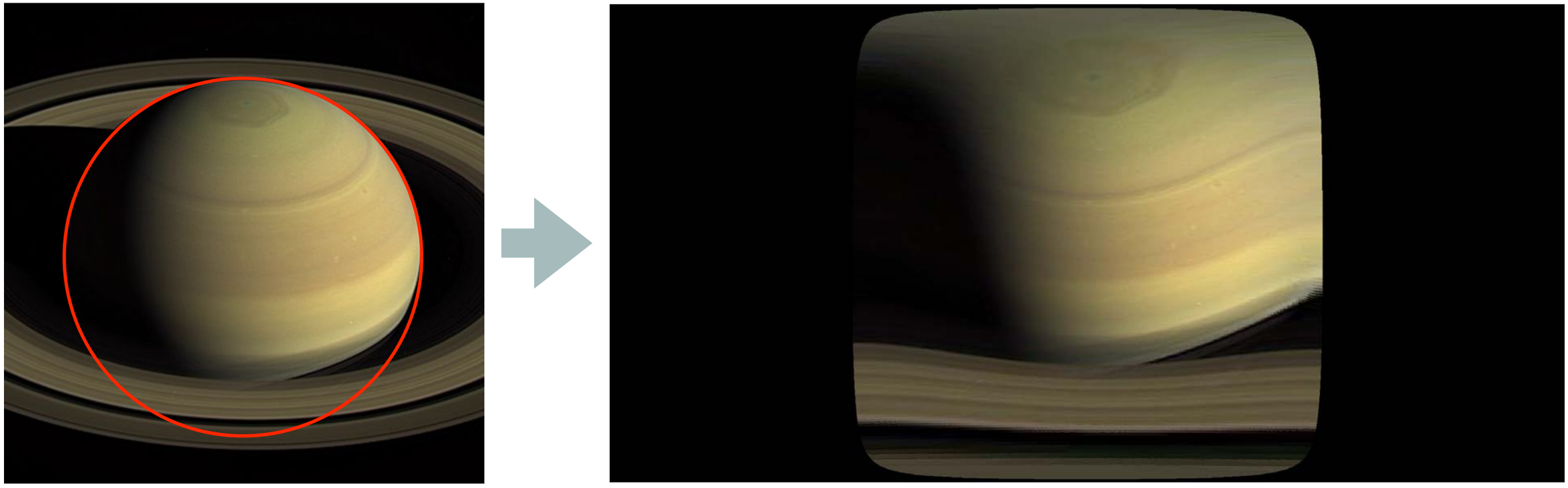
```
Dagikフォルダ/
  ├Dagik_Earth.exe（Windows版実行ファイル：64bitで動作させる場合は
  │                 Dagik_Earth_64bit.exeを実行します）
  ├Dagik_Earth.app（Mac版実行ファイル：Mac_Dagik_Earth_app.zipを解凍します）
  ├index.html（WEB版実行ファイル）
  └data/
    ├conf/（設定ファイル）
    │  ├lang-xxxx.txt（言語ごとの設定）
    │  ├init_conf.txt（投影環境（ハードウェアなど）ごとの設定）
    │  ├dow_conf.txt（WEB版コンテンツ設定）
    │  └conf.txt（Windows/Mac版コンテンツ設定）
    └images/（イメージファイル）
      ├map/（地球画像フォルダ）
      └screen/（キャプション画像フォルダ）
```

実行ファイル

設定ファイル

地球画像の作成補助ツールDemic

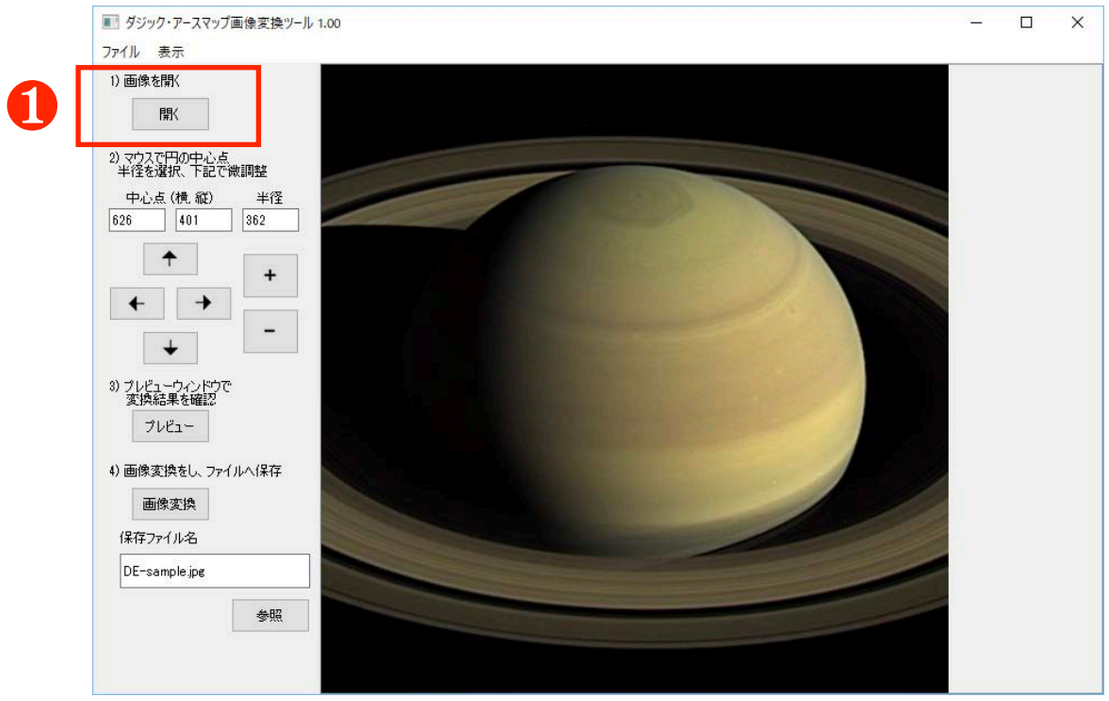
- Dagik Eath Map Image Convertorの略です。
- 写真上の円像をダジック・アース用正距円筒図法に変換するツールです。
  （詳細は2019年版ダジック・アースマニュアルp.73参照）
- ダジック・アースWEBのダウンロード＞画像変換ツールのページからダウンロードできます。（https://www.dagik.net/ダウンロード/画像変換ツール/）

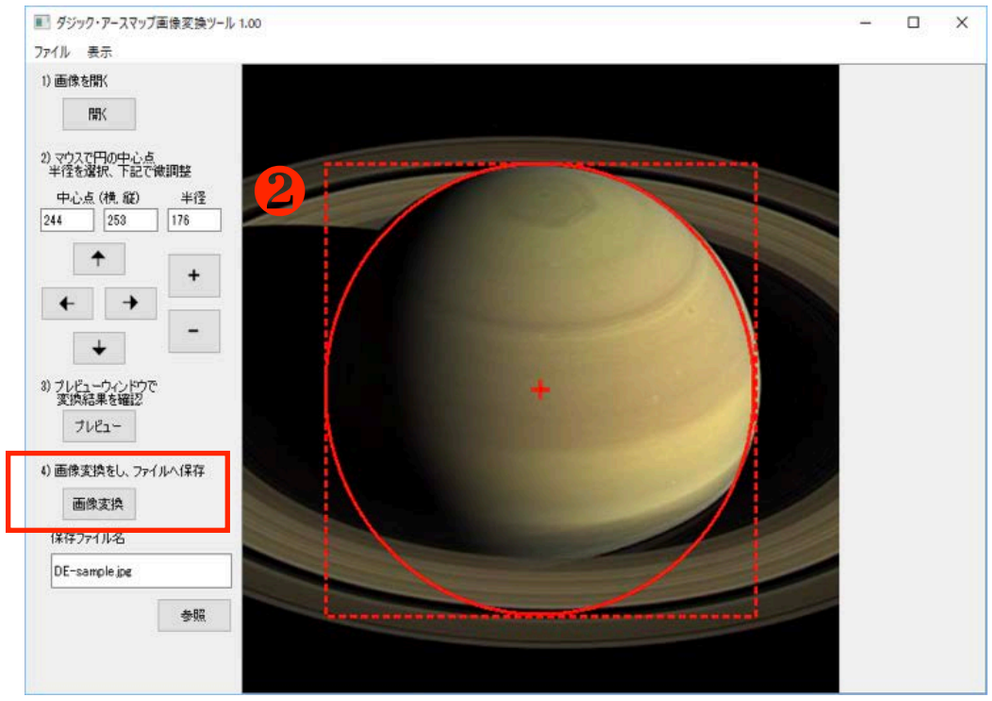


CASSINI 探査機による土星の撮影画像　(NASA / JPL-CALTECH / SPACE SCIENCE INSTITUTE HTTPS://WWW.JPL.NASA/GOV/VIDEO/DETAILS.PHP?ID=1441)

Demicの使い方

- 基本的な手順
  1. 画像を開く
  2. マウスで正距円筒図法に展開する円の位置を指定する
  3. 画像変換ボタンを押す





Demicの応用

➤ コマンドラインバッチ処理

各環境のターミナル（shell）で、実行プログラム指定に続いて以下のコマンドを入力します。

実行プログラム /batch <入力ファイル or 入力フォルダ>

または

実行プログラム /batch <入力ファイル or 入力フォルダ> <出力フォルダ> <設定iniファイル>

(新機能)

今回コマンドラインバッチに追加した新機能では、任意の設定iniファイルを指定できるようになりました。これにより複雑な変換設定をコマンドラインバッチ処理でも使い分けることができるようになりました。iniファイルの詳細については、Demic説明書の10pageをご覧ください。

WindowsのshellにおけるDemic実行プログラム指定

Demic.exe

MacのshellにおけるDemic実行プログラム指定
（従来のDemic.app open -a demic —argsによる指定はMac OS10.15で使えなくなりました）

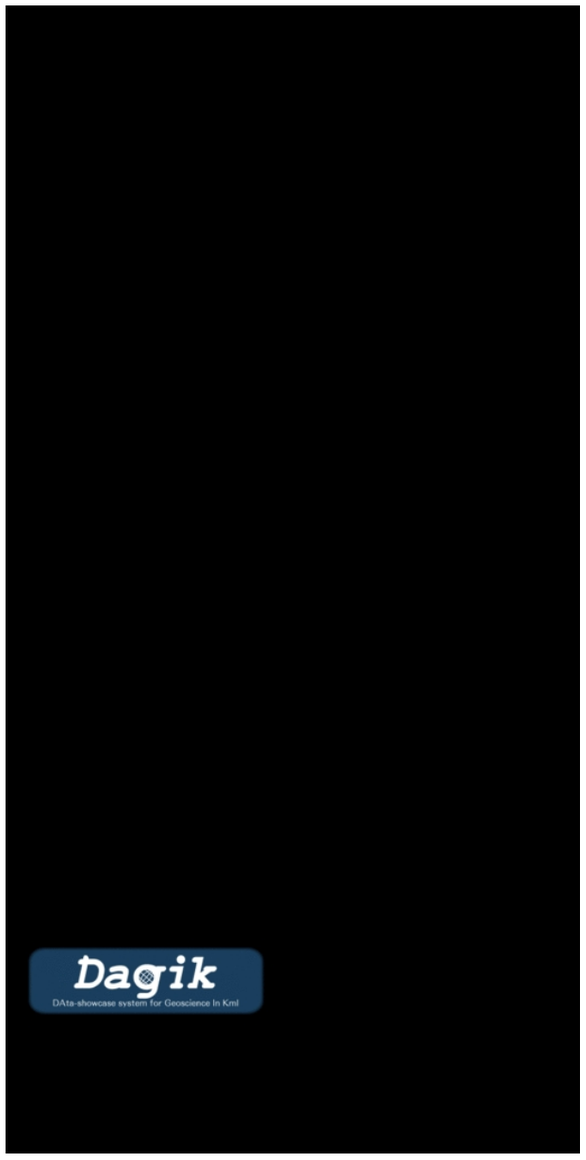
Demic.app/Contents/MacOS/demic

LinuxのshellにおけるDemic実行プログラム指定

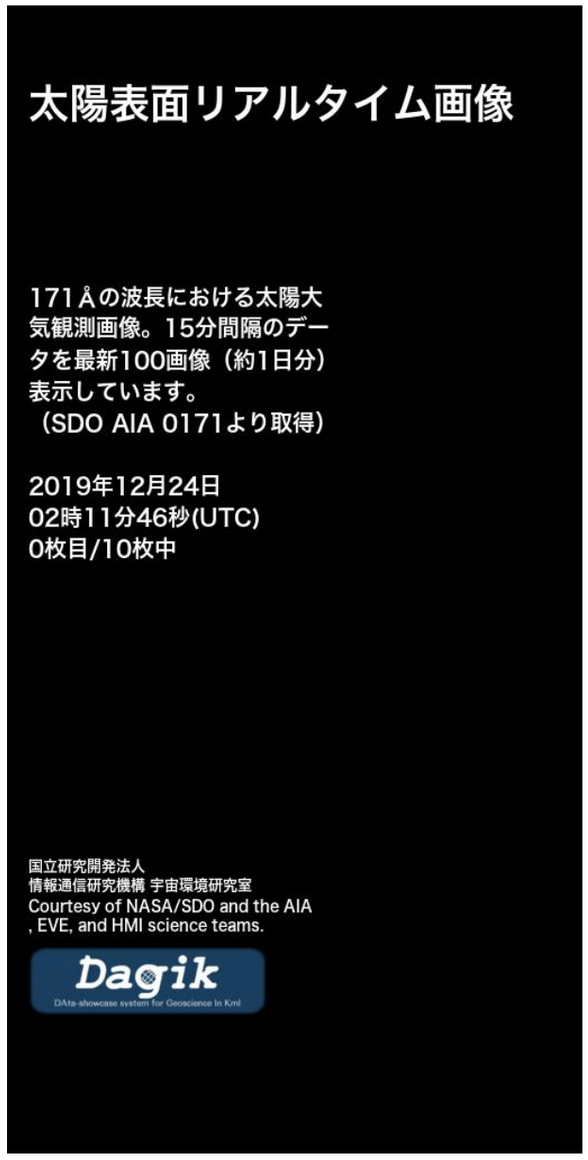
./Demic.AppImage

キャプション画像の作成補助ツールDesic

- Dagik Eath Screen Image Conposerの略です。
- ベースとなるキャプション画像に文字を追加して、キャプション画像を作成できます。
- ファイル名から日時などの情報を読み取り、自動的にキャプション画像を作成する機能を持ちます。



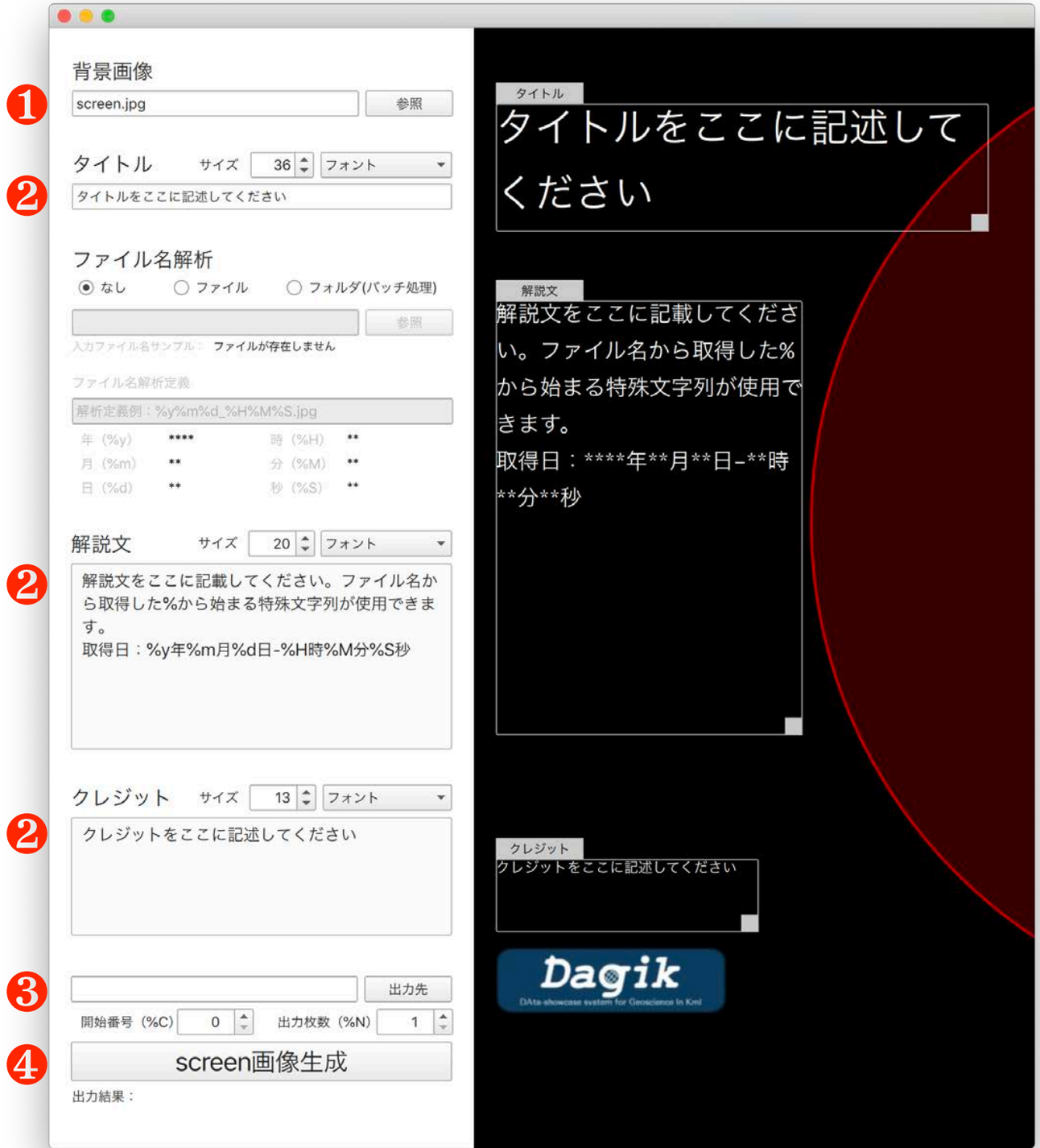
キャプションベース画像



出力画像

Desicの使い方（基本）

1. 任意の背景画像を使いたい時は、背景画像を指定します。
2. タイトル、解説文、クレジットの各文書を入力します。
3. 出力先のフォルダを指定します。
4. 「screen画像生成」ボタンを押します。



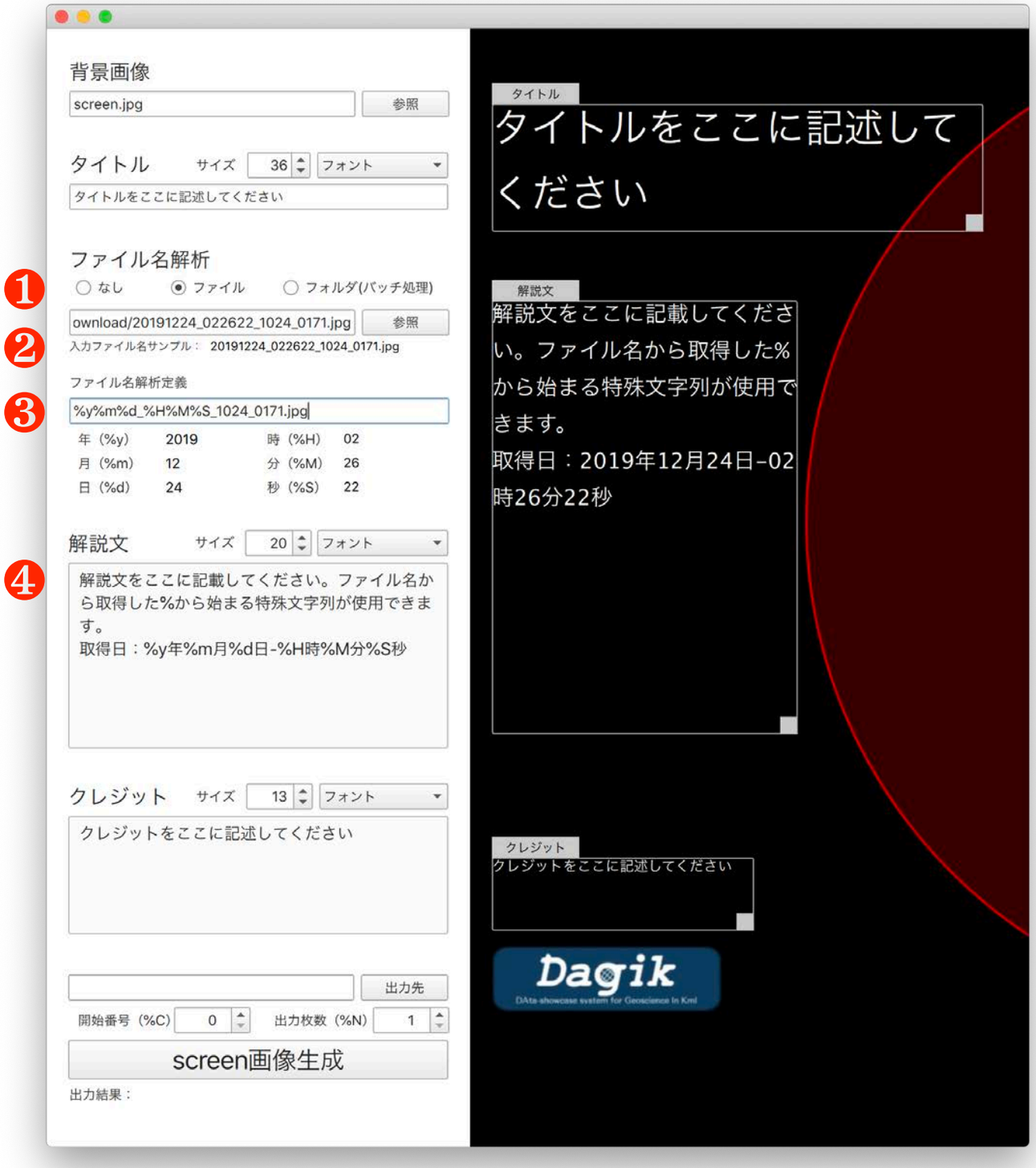
設定欄　　出力イメージ編集欄

Desicの使い方（ファイル名解析）

ファイル名解析をする場合は、基本操作に加えて、以下の指定を行います。

1. ファイル名解析のところで、「ファイル」または「フォルダ(バッチ処理)」を選択します。
2. ファイル名解析をする画像ファイルや画像ファイルを含むフォルダを指定します。
3. 「入力ファイル名サンプル」の表示を参考にしながら、ファイル名解析定義に、対応する年日時分秒の構成要素(%y,%m, %d,%H,%M,%S)がどこに入るかを代替文字で指定します。
4. 解説文に、年月日時分秒の代替文字を入力すると、その場所に、ファイル名解析から得られた年月日時分秒のデータが入ります。

代替文字には年月日時分秒以外に出力連番（%C）や出力枚数（%N）なども使えます



設定欄　　出力イメージ編集欄

Desicの使い方（コマンドラインバッチ処理）

- コマンドラインバッチ処理（予定）
  各環境のターミナル（shell）で、実行プログラム指定に続いて以下のコマンドを入力します。

実行プログラム <入力ファイル or 入力フォルダ>

または

実行プログラム <入力ファイル or 入力フォルダ> <出力フォルダ>

または

実行プログラム <入力ファイル or 入力フォルダ> <出力フォルダ> <設定iniファイル>

現在のところ開発途中で、パッケージアプリ化していないため、右記の実行プログラム指定は予定となります。

今回は開発言語であるpythonファイル（Desic/main.py）にバッチ処理のコマンド入力をしています。

WindowsのshellにおけるDesic実行プログラム指定

Desic.exe

MacのshellにおけるDesic実行プログラム指定
（従来のDemic.app open -a demic —argsによる指定はMac OS10.15で使えなくなりました）

Desic.app/Contents/MacOS/demic

LinuxのshellにおけるDesic実行プログラム指定

./Desic.AppImage

Demi,Desicを利用したリアルタイム処理（流れ）

1. ダジック・アースファイル一式を用意します。

2. リアルタイム画像がアップされているURLから、最新画像ファイルをダウンロードします。（60コマの動画にしたい場合は、常に最新から60枚分の画像がダウンロードフォルダに溜められておくようにします）

3. 入力フォルダにダウンロードフォルダを、出力フォルダにダジック・アースファイル一式のmapフォルダを指定し、Demicでバッチ処理します。

4. 入力フォルダにダウンロードフォルダを、出力フォルダにダジック・アースファイル一式のscreenフォルダを指定し、Desicでバッチ処理します。

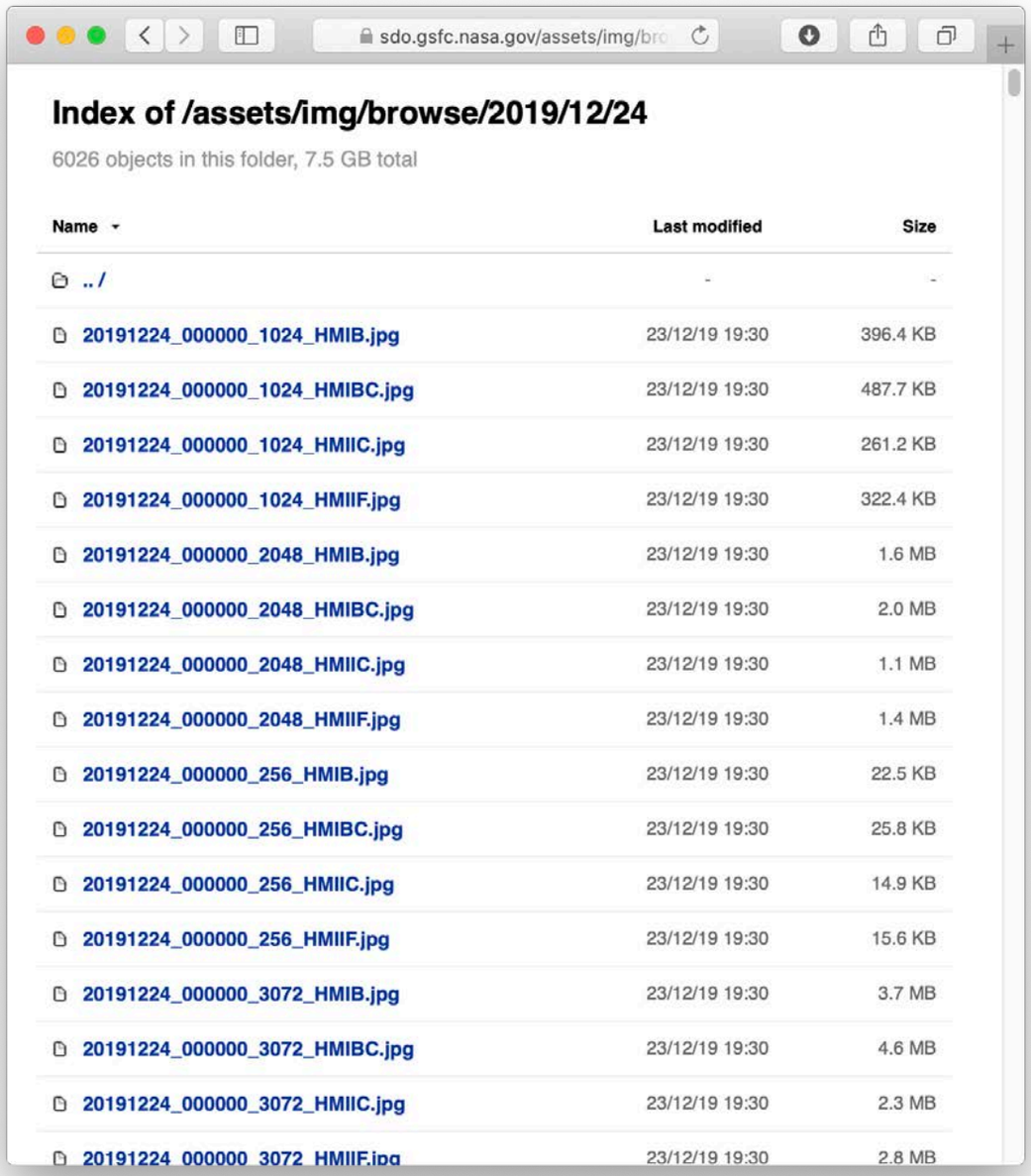
5. 2〜4の工程を実行するプログラムを、cronなどで定期実行します。

最新太陽画像のダウンロード例

- 最新太陽画像を蓄積しているNASAのSDOページ（https://sdo.gsfc.nasa.gov/assets/img/browse/）から、ファイルをダウンロードします。

- 上記ページでは、年月日別にディレクトリーが分けられて、同じ日付のディレクトリには、観測波長、画像サイズ、観測時刻で分けられたファイルがまとめて収納されています。

- ファイルは、
  年月日_時分秒_画像サイズ_観測波長.jpg
  の命名規則になっていることがわかります。今回はAIA171の観測波長の1024サイズのファイルをダウンロードしたいので、
  年月日_時分秒_1024_AIA171.jpg
  のファイルを検索してダウンロードすることになります。



最新太陽画像のダウンロード&Demic&Desic実行プログラム

- SDOから最新太陽画像をダウンロードし、DemicとDesicでバッチ処理させるプログラム例が以下のようになります。（pythonプログラム）

```
# This Python file uses the following encoding: utf-8
import os, re, urllib.request, datetime
from http.client import RemoteDisconnected
from bs4 import BeautifulSoup

#初期設定
downURL = "https://sdo.gsfc.nasa.gov/assets/img/browse/"
targetFILE = "_1024_0171.jpg"
fileMax = 10 #ファイル数
downDir ="/Users/iwashiro/download"

#ダウンロード実行関数
def download(downURL, targetFILE, downDir, fileMax):

  #downloadフォルダのファイルリストを取得
  downlist = sorted(os.listdir(downDir), reverse=True)

  #現在日時取得（UTC）
  dt=datetime.datetime.utcnow().date()
  print(dt)

  loadNO = 0
  while loadNO < fileMax:
    YY=dt.year
    MM=dt.month
    DD=dt.day
    dt = dt - datetime.timedelta(days=1)

    #ダウンロードURLを年月日に応じて修正
    URL = downURL + "{0:02d}".format(YY) + "/" + "{0:02d}".format(MM) + "/" +
"{0:02d}".format(DD) + "/"

    #URLリストを取得
    print("URLリスト取得中： " + URL)
    try: html = urllib.request.urlopen(URL)
    except (http.client.IncompleteRead) as e:
      page = e.partial
      print("URL取得が失敗しました")
```

```
    #対象のファイル名リストを取得（最新順）
    bsObj = BeautifulSoup(html, "html.parser")
    filelist= sorted(bsObj.find_all(string=re.compile(targetFILE)), reverse=True)

  #最新ファイルから順にダウンロードする。ただし最大ダウンロード数以内かつダウンロード内に無いファイル
    for link in filelist:
      if link not in downlist :
        print(str(loadNO+1) +".ロード実行： " + URL+link)
        try: urllib.request.urlretrieve(URL+link, downDir+"/"+link)
        except urllib.error.URLError as e:
          print("通信エラー： "+e.reason)
          print("不完全ファイルを削除します:"+link)
          os.remove(downDir+"/"+link)
        except RemoteDisconnected:
          print("通信が切断されました")
      else:
        print(str(loadNO+1) +".ファイル有： " + link)
      loadNO += 1
      if loadNO == fileMax:
        break


  #downloadフォルダのファイルリストを再取得
  downlist = sorted(os.listdir(downDir), reverse=True)

  for removefile in downlist[fileMax:]:
    print("削除： "+downDir+"/"+removefile)
    os.remove(downDir+"/"+removefile)

download(downURL, targetFILE, downDir, fileMax)

#desicを実行
os.chdir('/Users/iwashiro/Desic/')
os.system('python main.py /Users/iwashiro/download /Users/iwashiro/Dagik/data/images/screen')

#demicを実行
os.chdir('/Users/iwashiro/Demic/')
os.system('demic.app/Contents/MacOS/demic /batch /Users/iwashiro/download/')
```

Desicリリースとダウンロードプログラムの汎用化

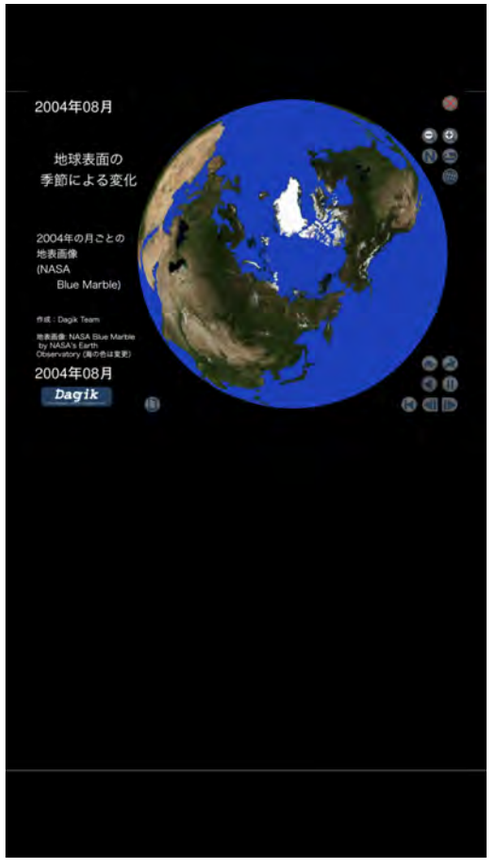
- 今回は、SDOからAIA171ファイルをダウンロード&Demic＆Desicを実行する専用のプログラムを別途記述しました。ダウンロード先に応じて毎回専用のプログラムを記述するのは大変ですので、汎用ダウンロードプログラム（Dagik Earth Realtime Image Downloder：Derid？）の開発も進めようと思っています。
- Desicは年度内に公開リリースを目指します。

電子書籍版ダジック・アースとその可能性

岩城邦典（国立研究開発法人情報通信研究機構）

ダジック・アース版電子書籍EPUB3サンプル

- サンプルとして「地表面の季節による変化」を組み込み
- ジャンル：land、コンテンツ名：Dagik_bluemarble
- 2019年版ダジック・アースマニュアル p.38



電子書籍とアプリの違い

電子書籍	アプリ
書籍データファイルと電子書籍プレーヤーが必要	アプリ単体で動作
WEB技術を使用しており、機能もWEB技術の制限に準じる	端末の持つハードウェアを最大限利用できる
OSに依存しない	OSに依存する

インタラクティブ対応電子書籍プレーヤー

- 数ある電子書籍プレーヤーの中でも、インタラクティブ操作に対応した電子書籍プレーヤーは限られています。

iOS/iPadOS/tvOS	iBooks
Android	Reasily（一部未対応あり）
Mac	iBooks
Win	Edge（一部未対応あり）
Linux	十分に動作するものは見つかっていません

電子書籍のファイルフォーマット

epub	世界標準フォーマット。 EPUB3ではインタラクティブな仕組みが組み込める。
ibooks	AppleのiBooks専用フォーマット。 インタラクティブな仕組みが組み込める。
AZW3、AZW、MOBI、Topaz、prc	AmazonのKindle専用フォーマット。 マルチメディアの再生までは組み込めるが、インタラクティブな仕組みは組み込めない。
book	ボイジャーの提唱する汎用フォーマット。 インタラクティブな仕組みは組み込めない。
zdf	シャープのGALAPAGOS専用フォーマット。インタラクティブな仕組みは組み込めない。

インタラクティブ電子書籍の商業流通網

iBooksストア	インタラクティブEPUB3可能。 インタラクティブiBooks可能。
Googleストア	インタラクティブEPUB3可能。
Kindleストア	インタラクティブEPUB3不可能。メディア再生の組み込みは可能。 AZW形式でもインタラクティブには対応しない。
その他	BCCKSなどはインタラクティブEPUB3の流通可能。 BOOTHなどは、本来決済代行システムではあるが、BOOTHの電子書籍コーナーにインタラクティブEPUB3を並べて流通させることは可能。 Stores.jp、BASEなどはショッピングカートサービスではあるが、これを利用し、独自の書店サイトを開設し流通させることが可能。

電子書籍のメリット

- OSに依存しないため、アプリのサポートが終了しても、読めなくなることはありません。
- 書籍のページ構成や順番は追加・編集が可能なので、任意のページを組み合わせて、オリジナル構成のストーリを持たせた電子書籍にカスタマイズすることが可能

ストーリー本サンプル

- Section1：オーロラとは?
  （独自編集ページ）
- Section2：オーロラを見てみましょう
  （ジャンル：space、コンテンツ名：Dagik_Syowa_ASC_2017-03-22、
  マニュアル：p.55）
- Section3：オーロラを上から見てみましょう
  （ジャンル：space、コンテンツ名：Dagik_ISSphoto_aurora、マニュアル：p.56）
- Section4：オーロラはどこに発生するのでしょう
  （ジャンル：space、コンテンツ名：Dagik_aurora_IMAGE_FUV、
  マニュアル：p.51）

電子書籍のページ構成カスタマイズ

- 手動で行うには、電子書籍の知識や膨大な手間が必要になるので、自動でそれらの処理を行える、電子書籍カスタマイズ編集システム「ECES（EPUB3 Constitution editing System）」を開発中。

将来の展開構想

- （ECES用ページ素材として可能な）ページ単位のEPUB3ダジックコンテンツを揃える。
- ECES（EPUB3 Constitution editing System）を公開し、カスタマイズ電子書籍を簡単に作れるようにすることで、学習性の高いダジック・アースコンテンツを作れると同時に、授業や講義、イベントなどでお土産として配布できるようにする。
- 既存の電子書籍ストアを通じて出版流通させることで、（世界へ）アウトリーチを広げる。
- 有料販売も行い、NPO法人ダジックアースの収入源の一つにする。