Genomics applications in the cloud with the DNAnexus Platform

Andrey Kislyuk

BOSC 2013
The DNAnexus Platform

Configurable cloud infrastructure for genomics
The DNAnexus Platform

Open and comprehensive API

Projects

Projects are special containers intended for collaboration; access permissions are controlled by the users who have ADMINISTER access to the project.

Projects vs Containers

Users can create projects, either through the website or using the API method `project/new`. Unless they are creating a project on behalf of an organization, they will be responsible for the storage costs of the data inside the project, as well as any compute time for any jobs run from the project. Projects have various metadata (name, description, tags, properties). The project also has two project-wide boolean flags.

- **PROTECTED**: If set to true, only members with "ADMINISTER" permission level can delete objects. Otherwise, members with "CONTRIBUTE" permission level can also delete objects from the project. To aid in the reproducibility of results from running apps in the project, project caches are not persistent and are recreated for each time an app is run.
- **RESTRICTED**: If set to true, the objects of this project cannot be cloned, and any running applet that reads from this project cannot write results to any other project (see later discussion for what this means, too).

In contrast, users cannot create generic data containers directly. Containers are generated automatically for different purposes involved in running an analysis, sharing data, etc. and will always be associated with a project or user/organization that will be responsible for its storage costs.

Project Metadata

Projects can have names, descriptions, properties, and tags. Properties may automatically set by the website front-end and are used for facilitating user experience.

List of API Methods

Project API Methods

The following methods are specified on this page:

- `/project/new`
- `/project-foo/addTags`
The DNAnexus Platform

Open and comprehensive SDK

Linux

OS X

Windows

Upload agents (Dropbox-like)
Batteries are included
The DNAnexus Platform

Powerful visualization tools built in
The DNAnexus Platform

Powerful visualization tools built in
The DNAnexus Platform

Quality control reports
Your visualization engine here

(there's an API for that)
Yes, we have Circos
DNAnexus is ready for clinical data
Secure, compliant, audited

Data compliance in accordance with:

- HIPAA
- GCP
- CLIA

- 21 CFR Part 11
- 21 CFR Part 58
- 21 CFR Part 493

European Data Privacy laws and regulations
Those are not just acronyms...

- All data encrypted with full-disk AES-256 at rest, SSL on the move
- Production access controls
- Third-party security audits
- Optional 2-Factor Auth
- LXC (Linux Containers) hypervisor
- Auditable by user
Your data is yours

We will **never** hold your data hostage

– Always exportable
– Always downloadable
– We’re not allowed to look at it
The DNAnexus Platform

Reliability

• Triple data redundancy
• Geographically distributed
• Job-level hardware fault tolerance
• Reproducible and auditable results for 6+ years

DNAnexus is ready for clinical data
Blazing fast development
Open-source stack

App wizard
walks you through app creation

Learn by example
fork our repos

Collaborate
deploy apps from GitHub
Debug quickly

- SDK tools for debugging
- Jobs start in 5 seconds under most circumstances
- Real-time job logs
Data standards

- **GTable**: Columnar indexed data store for genome data – Suitable for storing SAM, GFF/GTF, BED, VCF, etc.

- **I/O specifications**: Contracts between apps enable composition

- **Data type definitions and validators**: Foundations for Interoperability
Massive on-demand compute
<table>
<thead>
<tr>
<th>Task</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACXX-6-ID06</td>
<td></td>
</tr>
<tr>
<td>FastQC_v0.10.1</td>
<td>2h 42m 15s</td>
</tr>
<tr>
<td>parallel_bwa_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>fastq_splitter</td>
<td>1h 54m 52s</td>
</tr>
<tr>
<td>fastq_splitter</td>
<td>1h 30m 21s</td>
</tr>
<tr>
<td>parallel_bwa_v0.6.2:bwa_controller</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>bwa_aligner_v0.6.2</td>
<td></td>
</tr>
<tr>
<td>picard_merge_sam_files</td>
<td></td>
</tr>
<tr>
<td>FixMateInformation_Picard_v1.88</td>
<td></td>
</tr>
<tr>
<td>picard_mark_duplicates</td>
<td></td>
</tr>
<tr>
<td>CaptureStats_v2</td>
<td></td>
</tr>
<tr>
<td>BAMAnalyzerFIX</td>
<td></td>
</tr>
<tr>
<td>samtools_v0.1.9_index</td>
<td></td>
</tr>
<tr>
<td>snakemake</td>
<td></td>
</tr>
</tbody>
</table>
Instant-on supercomputer at your fingertips...
...only when you need it
Spin up thousands of instances
Specify instance types
Pay by the second
Community Collaboration
DNA-nexus Answers

A community working together
Instant collaboration

• Eliminate data transfer headaches
• Collaborate on data, tools, workflows in one environment
• Enable bioinformatics experts to deliver tools to biologists
So how do I add my app?
1. Run App Wizard
So how do I add my app?

DNAanexus App Wizard, API v1.0.0

Basic Metadata

Please enter basic metadata fields that will be used to describe your app. Optional fields are denoted by options with square brackets. At the end of this wizard, the files necessary for building your app will be generated from the answers you provide.

The name of your app must be unique on the DNAanexus platform. After creating your app for the first time, you will be able to publish new versions using the same app name. App names are restricted to alphanumeric characters (a-z, A-Z, 0-9), and the characters ".", "_", and "-".

App Name: spades

The title, if provided, is what is shown as the name of your app on the website. It can be any valid UTF-8 string.

Title []: SPAdes

The summary of your app is a short phrase or one-line description of what your app does. It can be any UTF-8 human-readable string.

Summary []: SPAdes assembler

The description of your app is a longer piece of text describing your app. It can be any UTF-8 human-readable string, and it will be interpreted using Markdown (see http://daringfireball.net/projects/markdown/syntax/ for more details).

Description []:
app spec
"name": "spades2",
"title": "Spades",
"dxapi": "1.0.0",
"version": "0.0.1",
"inputSpec": [
  {
    "name": "forward_reads",
    "class": "file",
    "optional": false
  },
  {
    "name": "reverse_reads",
    "class": "file",
    "optional": false
  }
],
"outputSpec": [
  {
    "name": "assembly",
    "class": "file"
  }
],
"runSpec": {
  "interpreter": "bash",
  "file": "src/spades2.sh"}
2. Add entry point code
#!/bin/bash

# spades2 0.0.1

# Generated by dx-app-wizard.

# Basic execution pattern: Your app will run on a single machine from
# beginning to end.

# Your job's input variables (if any) will be loaded as environment
# variables before this script runs. Any array inputs will be loaded
# as bash arrays.

# Any code outside of main() (or any entry point you may add) is
# ALWAYS executed, followed by running the entry point itself.

# See https://wiki.dnanexus.com/Developer-Portal for tutorials on how
# to modify this file.

main() {
    echo "Value of forward_reads: '\$forward_reads'"
    echo "Value of reverse_reads: '\$reverse_reads'"

    # The following line(s) use the dx command-line tool to download your file
    # inputs to the local file system using variable names for the filenames. To
    # recover the original filenames, you can use the output of "dx describe
    # "'$variable'" --name".

    dx download "$forward_reads" -o forward_reads
    dx download "$reverse_reads" -o reverse_reads

    # Fill in your application code here.

    # To report any recognized errors in the correct format in
    # $HOME/job_error.json and exit this script, you can use the
# Fill in your application code here.
#
# To report any recognized errors in the correct format in
# $HOME/job_error.json and exit this script, you can use the
# dx-jobutil-report-error utility as follows:
# dx-jobutil-report-error "My error message"
#
# Note however that this entire bash script is executed with -e
# when running in the cloud, so any line which returns a nonzero
# exit code will prematurely exit the script; if no error was
# reported in the job_error.json file, then the failure reason
# will be AppInternalError with a generic error message.

... spades.py -1 forward_reads -2 reverse_reads -o assembly

  tar -cf assembly.tar.gz assembly

  # The following line(s) use the dx command-line tool to upload your file
  # outputs after you have created them on the local file system. It assumes
  # that you have used the output field name for the filename for each output,
  # but you can change that behavior to suit your needs. Run "dx upload -h"
  # to see more options to set metadata.

  assembly=$(dx upload assembly.tar.gz --brief)

  # The following line(s) use the utility dx-jobutil-add-output to format and
  # add output variables to your job's output as appropriate for the output
  # class. Run "dx-jobutil-add-output -h" for more information on what it
  # does.

  dx-jobutil-add-output assembly "$assembly" --class=file
3. Build, test, publish
WARNING: dxpy:app is missing a summary, please add one in the "summary" field of dxapp.json

Created temporary project project-B6YGgxkJYY8XbX507GYQ00k7 to build in

DEBUG: dxpy: Building in /Users/kislyuk/Desktop/projects/spades
DEBUG: dxpy: Uploading in spades

Created applet applet-B6YGjJVJYY8XbX507GYQ00kK successfully
Will create app with spec: {u'name': u'spades', u'title': u'Spades', u'outputSpec': [{u'name': u'assembly', u'class': u'file'}], u'runSpec': {u'interpreter': u'bash', u'file': u'src/spades2.sh'}, u'version': u'0.0.1', u'inputSpec': [{u'optional': False, u'name': u'forward_reads', u'class': u'file'}, {u'optional': False, u'name': u'reverse_reads', u'class': u'file'}], u'dxapi': u'1.0.0'}

Attempting to create version 0.0.1...
App spades/0.0.1 does not yet exist
Created app app-B6YGjJb333PXbX507GYQ00kQ
Uploaded app spades/0.0.1 (app-B6YGjJb333PXbX507GYQ00kQ) successfully
You can publish this app with:
  dx api app-spades/0.0.1 publish "{\"makeDefault\": true}"

kislyuk@aurora:~/Desktop/projects>
done
SPAdes Genome Assembler

Inputs
- forward_reads: file
- reverse_reads: file

Outputs
- assembly: file

Run this app from the command line:
$ dx run spades
# For help specifying inputs:
$ dx run spades -h
To get dx, download the Platform SDK.

Pricing
Compute cost (variable)

Permissions
No special permissions

Latest Update
Version
0.0.1+build.20130530.0703 May 30, 2013

Added by
Andrey Kislyuk
Run Analysis for SPAdes

View job progress in your project's Monitor tab.

Inputs

- forward_reads
- reverse_reads

App

- SPAdes
  - set inputs

Outputs

- assembly

Add a Step

Close
Maps reads to a reference genome using the Burrows-Wheeler Aligner

Burrows-Wheeler Aligner (BWA) is an efficient program that aligns relatively short nucleotide sequences against a long reference sequence such as the human genome. This app runs BWA to map `letterspace` reads to a reference genome and produce mappings.

**Inputs:**

*Reads:* An array of Reads table objects that will be mapped to the reference genome. If more than one Reads object is provided, the results are combined into a single Mappings output.

*Output name:* The name of the resulting Mappings table object (optional; if not provided, the name will be based on the Reads name).

*Reference genome:* The genome that the reads will be mapped against. BWA requires a special processing on the genome, called indexing; this processing can take several hours for long genomes. If the genome given in the input is not indexed for BWA, the app will automatically index it and include an indexed version in the output, for future use. When possible, please run this app with an indexed genome to avoid re-indexing. DNAanexus provides several pre-indexed genomes in the 'Reference Genomes' public project.

*Mapping algorithm:* BWA implements two different algorithms, both based on Burrows-Wheeler Transform (BWT). The first algorithm, called ‘alin’ is designed for short queries up to ~200bp with low error rate (<3%). It does gapped global alignment w.r.t. queries, supports paired-end reads, and is one of the fastest short read alignment algorithms to date while also visiting suboptimal hits. The second algorithm, called ‘bwasw’, is designed for long reads with more errors. It performs heuristic Smith-Waterman-like alignment to find high-scoring local hits (and thus chimeras). On low-error short queries, 'bwasw' is slower but less accurate than the first algorithm, but on long queries, it is better. Using a value of ‘auto’ will automatically choose a suitable algorithm based on the length of the reads in the
Reads per chunk: This app parallelizes itself by dividing the input into chunks of a certain size (a certain number of reads), and mapping each chunk individually. Lower chunk sizes lead to higher levels of parallelization, reducing the wall-clock time that one has to wait for the app to finish. However, lower chunks sizes may also increase the cost of running the app in the cloud, as they lead to a higher number of chunks, each of which adds a constant processing overhead. The default value is 25 million reads per chunk. DNA Nexus suggests caution when experimenting with this parameter.

Discard unmapped rows?: If selected, unmapped reads will not be included in the Mappings output.

Low-level parameters: Users familiar with the BWA executable can directly manipulate the parameters that are used for the bwa aln, bwa samse, bwa samse and bwa bsw calls. These parameters are: aln_n, aln_o, aln_e, aln_l, aln_d, aln_i, aln_k, aln_m, aln_0, aln_e, aln_R, aln_q, aln_N, sampe_a, sampe_o, sampe_n, sampe_N, sampe_c, sampe_s, sampe_n, sw_a, sw_b, sw_q, sw_r, sw_w, sw_m, sw_T, sw_c, sw_z, sw_s, sw_N. Each one of these parameters directly correspond to the respective command-line argument, e.g. aln_o corresponds to the -o option of bwa aln (maximum number of gap opens). Certain options, such as the -t option of bwa aln, are not exposed to users because they are set by the app, based on the kind of cloud environment that the app runs on.

Outputs:

Mappings: The mappings produced by BWA are output in Mappings table object. (Developers can look at http://wiki.dnanexus.com/Types/Mappings for more information). Mappings objects can be then used as inputs to certain variation calling apps, mappings QC apps, etc.; they can also be visualized in the genome browser.

Indexed reference genome: An indexed version of the reference genome, for future use as input to this app. As mentioned earlier, if the app is given a reference genome that is not indexed for BWA, it will index it. This output contains the indexed version so that you can provide it as input to future invocations of the app.

CATEGORIES
Read Mapping

SOURCE
Browse source code
Fork this app: git clone git@github.com:dnanexus/bwa.git

CITATIONS
Reproducibility

Ever try to reproduce results from a bioinformatics paper?
How about CLIA compliance?

All objects are versioned
Analysis I/O is read-only
Jobs enter into project’s permanent record
Publishing

Authors who publish their software

• Don’t worry about supporting diverse installs
  – You installed my package on WHAT?

• Leverage all Platform features
  – Accessible UI

• Compose with other apps
  – It’s an ecosystem
  – Publish your workflows as apps, too
Recognition

Users are encouraged to cite app authors

One-click bibliographies coming soon
We care about developers
DNAnexus developer program

• $1000 credit
• App Bounties
• Featured Apps
Vision
DNAnexus roadmap

• AWS Glacier – *cheap data archival*

• Projects as publications

  *Supplementary materials!*

And of course...

• More apps
Always improving...
...And so are our cloud providers
We pass on the savings to you
DNAnexus is the platform for publishing your algorithms
DNAnexus is the platform for delivering genomics results to users
Acknowledgments

*DNAnexus is*
Andreas Sundquist
Arend Sidow
Serafim Batzoglou

Major Investors

[Google Ventures](#)
[TPG Biotech](#)
[First Round Capital](#)
Thank you

dnanexus.com