Outline

a. Introduction
b. Data objects & object adaptors
c. Ensembl documentation
d. The Registry & Ensembl API script design
e. Coordinate systems & slices
f. Features
g. Genes, transcripts, exons & translations
h. External references
Ensembl stores features and DNA sequence in a number of coordinate systems.

**Top level coordinate system**

**Chromosome**

**Contigs**

**Clones**

(Tiling path)

**Sequence level coordinate system**
Regions in one coordinate system may be constructed from a tiling path of regions from another coordinate system.
# Obtain all coordinate systems for human

use Bio::EnsEMBL::Registry;
my $registry = 'Bio::EnsEMBL::Registry';

$registry->load_registry_from_db(
    -host => 'ensembldb.ensembl.org',
    -user => 'anonymous'
);
my $coordsystem_adaptor = $reg->get_adaptor( 'Human', 'Core', 'CoordSystem' );

my $coordsystems = $coordsystem_adaptor->fetch_all;

while ( my $coordsystem = shift @{$coordsystems} ){
    print $coordsystem->name, "\t",
    $coordsystem->version, "\t",
    $coordsystem->rank ,"\n";
}

Note use of ‘while’ and ‘shift’ instead of ‘foreach’ – more memory efficient way for large datasets
### Coordinate Systems – Code Output

**OUTPUT:**

<table>
<thead>
<tr>
<th>name</th>
<th>version</th>
<th>rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>chromosome</td>
<td>GRCh37</td>
<td>1</td>
</tr>
<tr>
<td>supercontig</td>
<td>GRCh37</td>
<td>2</td>
</tr>
<tr>
<td>clone</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>contig</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>chromosome</td>
<td>NCBI36</td>
<td>5</td>
</tr>
<tr>
<td>chromosome</td>
<td>NCBI35</td>
<td>6</td>
</tr>
<tr>
<td>chromosome</td>
<td>NCBI34</td>
<td>7</td>
</tr>
<tr>
<td>lrg</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

- **Latest assembly, top level**
- **Latest assembly, sequence level**
- **Old assemblies**, used for mapping features between assembly versions.
- **Locus-Reference Genes**, used in clinical tests.
Slices

A Slice Data Object represents an arbitrary region of a genome, a slice of a Sequence Region.

Slices are not directly stored in the database.

A Slice is used to request sequence or features from a specific region in a specific coordinate system.
Slices - Code Example (1)

# get a slice covering the entire human Y chromosome

my $slice_adaptor = $registry->get_adaptor( 'Human', 'Core', 'Slice' );
my $slice = $slice_adaptor->fetch_by_region( 'chromosome', 'Y' );
print "Coord system:\t", $slice->coord_system_name, "\n",
    "Seq region:\t", $slice->seq_region_name, "\n",
    "Start:\t", $slice->start,"\n",
    "End:\t", $slice->end,"\n",
    "Strand:\t", $slice->strand,"\n",
    "Slice:\t", $slice->name, "\n";

OUTPUT:
Coord system: chromosome
Seq region: Y
Start: 1
End: 59373566
Strand: 1
Slice: chromosome:GRCh37:Y:1:59373566:1
# get the slice adaptor
$slice_adaptor = $reg->get_adaptor('human', 'core', 'slice');

# fetch a slice on a region of chromosome 12
$slice = $slice_adaptor->fetch_by_region('chromosome', '12', 1e6, 2e6);

# print out the sequence from this region
print $slice->seq, "\n";

# get all clones in the database and print out their names
@slices = @{$slice_adaptor->fetch_all('clone')};
foreach $slice (@slices) {
    print $slice->seq_region_name, "\n";
}
Exercise 3

(a) Fetch all chromosomes for human. Determine their number and print the name and length for each of them. The number of chromosomes is probably not what you would expect! Why is this?

(b) Use the gene stable id 'ENSG00000101266' to fetch a slice surrounding this gene with 2kb of flanking sequence.
   (a) hint: use the Ensembl API documentation to find an appropriate method in SliceAdaptor class which retrieves a slice given a gene stable id (pay attention to the method’s arguments)

(c) Fetch the sequence of the first 10MB of chromosome 20 and write it to a file in FASTA format. Print the number of genes in this region.
   ✓ hint: Slice objects inherit from Bio::Seq so can be written to file easily using Bio::SeqIO, e.g.:
   
   my $output = Bio::SeqIO->new( -file=>'>filename.fasta',
                                -format=>'Fasta');
   $output->write_seq($slice);