PROTEGE API	OWL API	JENA API
extension of the OWL API Protégé is also an open-source, Java tool that provides an extensible architecture for the creation of customized knowledge-based applications.	The OWL API bypasses RDF to provide services based on OWL. **It is not RDF-friendly** and you won't be applying SPARQL queries any time soon.	**Most Flexible** one as it covers all of RDF and therefore can be used to create OWL constructs, axioms and run inferences.
the Protege-OWL API does *not* sit on top of Jena. It only uses Jena for parsing and provides a Jena "view" (implementation of the Graph interface so that some Jena services can be exploited for Protege as well). But you are right - the Protege-OWL API is good for newcomers, assuming they understand the layering on top of the core Protege API: no methods that are overloaded as deprecated should be used.	The OWL API is a Java API and reference implmentation for creating, manipulating and serialising OWL Ontologies. The latest version of the API is focused towards OWL 2 The OWL API is open source and is available under either the LGPL or Apache Licenses The OWL API includes the following components: • An API for OWL 2 and an efficient in-memory reference implementation • RDF/XML parser and writer • OWL/XML parser and writer • OWL Functional Syntax parser and writer • Turtle parser and writer • KRSS parser • OBO Flat file format parser • Reasoner interfaces for working with reasoners such as FaCT++, HermiT, Pellet and Racer	. Jena is a general purpose RDF API (that means RDF data, not just ontologies) plus an OWL API, plus SPARQL processor, reasoning support, pluggable database backends and various assorted external tools like the Eyeball data validator and Joseki server. A number of groups have built interesting tools that work with Jena such as the D2RQ database mapper. Jena is one of the most widely used Java APIs for RDF and OWL, providing services for model representation, parsing, database persistence, querying and some visualization tools. Protégé-OWL has always had a close relationship with Jena. The Jena ARP parser is still used in the Protégé-OWL parser, and various other services such as species validation and datatype handling have been reused from Jena. It was furthermore possible to convert a Protégé OWLModel into a Jena OntModel, to get a static snapshot of the model at run time. This model, however had to be rebuilt after each change in the model.
PROTEGE API	OWL API	Jena API

Complexity/easy to use - Like other Java APIs, the libraries are needed to be imported in the source code of the application. Provided with the standard Protege installation. - Protege API is the most complete, and has good compatibility with Protege - Protege API does not need any other installations - Protege API includes most of the Jena properties	Complexity/easy to use - higher level of abstraction - loading ontologies is easy, running SWRL more complex - loading ontologies, saving ontologies, entities, deleting Entities - straight forward java programming	Complexity/easy to use - easy/reasonable to use - write both java programs and also use command-line inputs
The Protégé Java API lets us control the ontology's internal representation. For example, we can create new classes and instances programmatically. Although we can use this API to manage all aspects of the Protégé internal representation, doing so is complicated because of the API's flexible but low-level nature. The tool has no performance engine, so we must use procedural Java code to access the API.	Reasoning, support for SWRL - own examples on how to create an ontology andd some rules - interaction with reasoning - Two reasoner implementations available: Pellet and FaCT++. - Pellet should support SWRL rules - UNIK uses OWL-API - OWL-API most probably easiest to use	