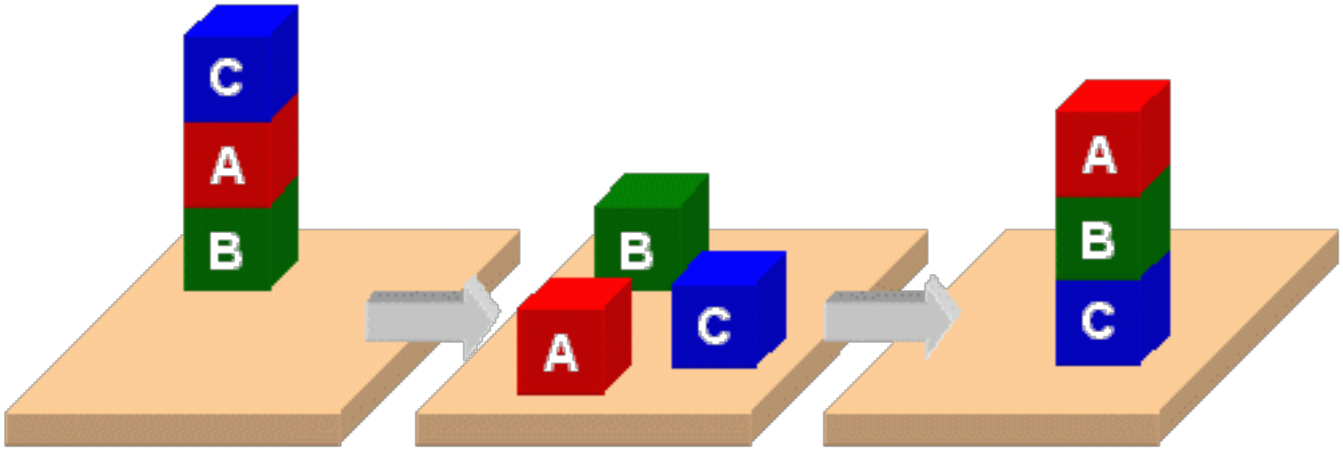
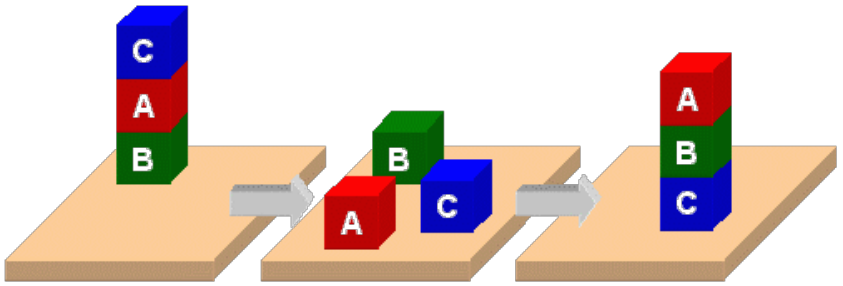
PDDL

Planning Domain Description Language



PDDL



- **Planning Domain Description Language**
- Based on STRIPS with various extensions
- First defined by Drew McDermott (Yale) et al.
  - Classic spec: PDDL 1.2; good reference guide
- Used in biennial International Planning Competition (IPC) series (1998-2022)
- Many planners use it as a standard input
- Latest version is 3.1 and newer variants exist

PDDL is still widely used

- After 22+ years, PDDL is still used in many planning systems and domains as a standard for input and output
- It's representation was updated, e.g. adding
  - fluents (facts that change over time)
  - preferences (aka soft constraints)
- New variants support multiple agents, ontologies, and more
- It still retains is traditional Lisp syntax

PDDL Representation

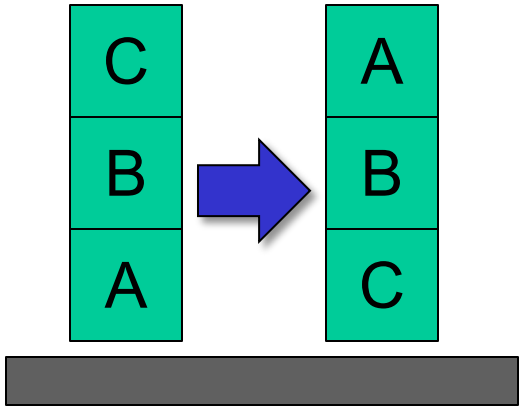
- Task specified via two files: **domain file** and **problem file**
  - Both use a logic-oriented notation with Lisp syntax
- **Domain file** defines a domain via *requirements*, *predicates*, *constants*, and *actions*
  - Used for many different problem files
- **Problem file:** defines problem by describing its *domain*, specific *objects*, *initial state* and *goal state*
- **Planner:** domain + problem → a plan

Blocks Word Domain File


```
(define (domain BW)
  (:requirements :strips)
  (:constants red green blue yellow small large)
  (:predicates (on ?x ?y) (on-table ?x) (color  ?x ?y) … (clear ?x))
  (:action pick-up
     :parameters (?obj1)
     :precondition (and (clear ?obj1) (on-table ?obj1)
                                (arm-empty))
     :effect (and (not (on-table ?obj1))
                  (not (clear ?obj1))
                  (not (arm-empty))
                  (holding ?obj1)))
  … more actions ...)
```

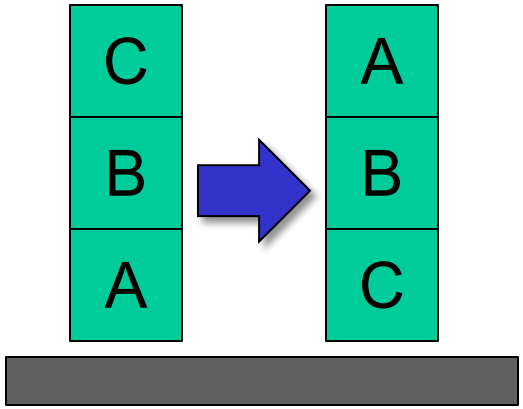
Blocks Word Problem File


```
(define (problem 00)
    (:domain BW)
    (:objects A B C)
    (:init (arm-empty)
            (ontable A)
           (on B A)
           (on C B)
           (clear C))
    (:goal (and (on A B)
                (on B C)
                (ontable C)))
```



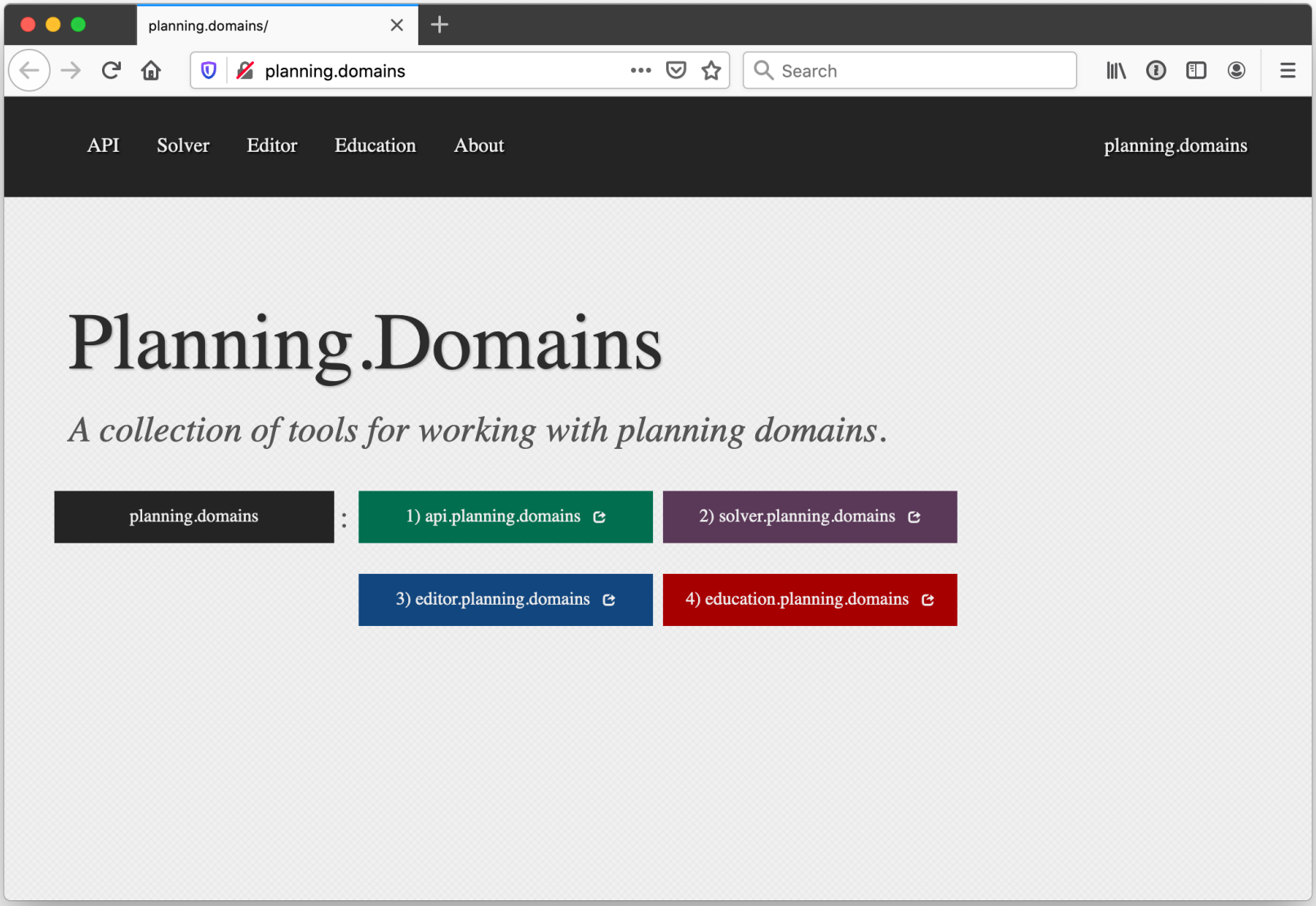
Blocks Word Problem File


```
(define (problem 00)
    (:domain BW)
    (:objects A B C)
    (:init (arm-empty)
           (on B A)
           (on C B)
           (clear C))
    (:goal (and (on A B)
                (on B C))))
```



Begin plan
1 (unstack c b)
2 (put-down c)
3 (unstack b a)
4 (stack b c)
5 (pick-up a)
6 (stack a b)
End plan

https://planning.domains/



Planning.domains

- Open source environment for providing planning services using PDDL (GitHub)
- Default planner is ff (aka, fastForward)
  - very successful forward-chaining heuristic search planner producing sequential plans
  - Can be configured to work with other planners
- Use interactively or call via web-based API
- We've used it for to extend blocks world domain in homework