Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

C	**BasicReceiptBuilder**	Minimal implementation of **ReceiptBuilder** for basic receipts
C	**Begonia**	Concrete implementation of a **Begonia** plant
C	**Cactus**	Concrete implementation of a **Cactus** plant
C	**CareStrategy**	Abstract class representing different care strategies for plants
C	**ChatIterator**	**Iterator** for chat messages
C	**Command**	
C	**ConcretePlantBuilder**	Concrete builder that composes multiple **Plant** instances into a single composite **Plant** according to a decorator parameter
C	**Customer**	Represents a customer in the system. Inherits from **Person** and implements behaviour for receiving messages sent through a **Section**
C	**DeadState**	Represents the dead state of a plant in the state pattern. In this state, the plant can no longer grow or respond to care
C	**DecorativePot**	Decorator that adds a decorative pot to a plant
C	**FlowerFactory**	Factory for creating flower-type plants
C	**GiftWrapDecorator**	Decorator that adds gift wrapping to a plant
C	**GiveSunlight**	
C	**Greenhouse**	A concrete subject that manages a collection of plants
C	**GreenhouseInterface**	An interface for the "Subject" in the Observer design pattern
C	**GroundStaff**	Represents ground staff in the system. Inherits from **Staff** and handles "Care" requests by executing watering and sunlight commands on specified plant tags. Also provides helper methods to add plants to a greenhouse and configure care commands
C	**HeavyCareStrategy**	Concrete class implementing the **CareStrategy** for plants needing heavy care. This class provides specific implementations for watering, sunlight, and soil care tailored to plants that require more intensive care
C	**HelpDesk**	A **HelpDesk** is a concrete **Section** that broadcasts messages to all participants and routes help requests to the admin or the first available **Staff** handler in the section. Behaviour mirrors **Section** derived types such as **SalesRoom** but is intended for assistance
C	**Inventory**	A singleton class that manages the inventory of plants in the nursery
C	**InventoryInterface**	An interface for the "Observer" in the Observer design pattern
C	**Iterator**	

C **LightCareStrategy** — Concrete class implementing the **CareStrategy** for plants needing light care. This class provides specific implementations for watering, sunlight, and soil care tailored to plants that require minimal care

C **Manager** — Represents a manager in the system. Inherits from **Staff** and handles requests that require managerial attention

C **Mango** — Concrete implementation of a **Mango** plant

C **MatureState** — Represents the mature state of a plant. This state handles the behavior of a plant that has fully grown and is now aging. It requires normal care and will eventually transition to the dead state

C **ModerateCareStrategy** — Concrete class implementing the **CareStrategy** for plants needing moderate care. This class provides specific implementations for watering, sunlight, and soil care tailored to plants that require balanced care

C **Oak** — Concrete implementation of an **Oak** plant

C **Person** — Abstract base class representing a person. Provides common attrributes and methods for customers and staff

C **Plant** — Abstract base class representing a plant in the nursery

C **PlantBuilder** — Interface for building composite **Plant** objects

C **PlantDecorator** — Base class for decorating plants with additional features

C **PlantFactory** — Factory class for plant creation

C **PlantGroup** — Composite class for grouping multiple plants

C **PlantInfo** — Holds plant information for receipt storage

C **PlantIterator** — **Iterator** for **Plant** objects

C **PlantState** — Abstract base class for plant states. This class defines the interface for handling state-specific behavior of plants. Each concrete state class will implement the handle method to define how the plant behaves in that particular state

C **Receipt** — Stores the items and total for a completed sale

C **ReceiptBuilder**

C **Rose** — Concrete implementation of a **Rose** plant

C **SalesAssistance**

C **SalesClerk** — Represents a sales clerk in the system. Inherits from **Staff** and handles Purchase requests and executes **SellPlant** commands

C **SalesRoom** — A **SalesRoom** is a concrete **Section** that broadcasts messages to all participants and routes requests to the admin or the first available **Staff** handler in the section

C **Section** — A **Section** manages a collection of **Person** instances, maintains a history of messages, and provides an interface to notify participants when a **Person** sends a message or makes a request. Concrete **Section** types implement the **notify(Person*)** method to define how messages are propagated Sections also provide an iterator for traversing the contained chat history

C **SeedState**   Represents the seed stage of a plant in its lifecycle. This state handles the transition from seed to sprout

C **SellPlant**   Concrete **Command** that handles selling plants and building receipts

C **SproutState**   Represents the sprout state of a plant in its lifecycle. This state handles the behavior and transitions specific to a sprout

C **Staff**   Represents a staff member in the system. Inherits from **Person** and provides request handling (Chain of Responsibility) and message receiving

C **SucculentFactory**   Factory for creating succulent-type plants

C **TreeFactory**   Factory for creating tree-type plants

C **Tulip**   Concrete implementation of a **Tulip** plant

C **WaterPlant**   Concrete **Command** that waters plant(s) in the greenhouse

Generated by **doxygen** 1.9.8