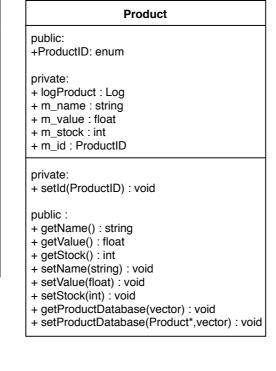
StateMachine VendingMachine protected: private: + currentState : unsigned char + m transactionCash : float + m interface : Interface* + productDatabase : Product* private: + m_maxStates : const unsigned char + logVendingMachine : Log + m eventGenerated : bool + m_pEventData : EventData* public: + cancelEvent(): void + cashIncrementEvent(float) : void protected: + productSelectionEvent(int): void + externalEvent(char,EventData*): void + internalEvent(char.EventData*): void private: + ST_ldle(EventData*) : void private: + ST Devolution(EventData*): void + StateEngine(): void + ST_Validation(VendingMachineData*): void + ST_Transaction(VendingMachineData*): void + ST_Deployment(VendingMachineData*): void



Interface

public:

+getSystemInput(SystemData*): virtual void +setSystemOutput(SystemData*): virtual void

+getUserInput(UserData*): virtual void +setUserOutput(UserData*): virtual void

+printAdvertising(AdvertisingData*): virtual void +insertAdvertising(AdvertisingData*): virtual void

Log public:

+ Level : enum

private:

m_logLevel : Level m_scope : string

public:

+ setLevel(Level) : void + setScope(string) : void

+ error(string): void + warn(string): void

+ info(string): void

+ debug(string): void

Node

private: + m nodeData : int

+ m nextNode : Node*

public: + getNodeData(): int

+ getNextNode(): Node* + setNodeData(int) : void

+ setNextNode(Node*) : void

Queue

private: + m_head : Node

public:

+ pop(): int + push(int) : void

DebugInterface

+ field: type

+ method(type): type

EventData

+ field: type

+ method(type): type

