用户操作

|  |  |
| --- | --- |
| 方法名 | auction |
| 说明 | 将一张物品牌进行拍卖  该用户目前不在拍卖状态中  向拍卖表中添加新数据 |
| 参数 | player\_id, card\_id,price |
| 返回值 | statuscode |
| 操作 | 1. 确认Player(player\_id).on\_sale == false 2. 确认Player\_Card表中有card\_id和player\_id的数据, 且is\_bought==false 3. 在Auctions表中添加新的数据，除一口价外，price=0；一口价price=price 4. 从Player\_Card表中删除数据 5. Player(player\_id).on\_sale设为true 6. 返回AUCTION\_SUCCESS/AUCTION\_FAILED\_ONSALE/ AUCTION\_ERROR |

|  |  |
| --- | --- |
| 方法名 | bid |
| 说明 | 对进行中的拍卖进行竞价  出价需低于所持金  明拍：不是目前最高出价人，出价比现价高。向原最高出价人退还出价，从新的最高出价人所持金中扣除出价。更新最高出价人和出价信息。向竞价表中添加新数据。  暗拍：没有对此物品出过价。从所持金中扣除出价。如果超过原最高出价人，则更新最高出价人和出价信息，但暂不退还出价。向竞价表中添加新数据。 |
| 参数 | player\_id, auction\_id, price |
| 返回值 | statuscode |
| 操作 | 1. 确认存在Auction(auction\_id) 2. 确认price <= Player(player\_id).gold    1. 如果是明拍, 确认price > Auction(auction\_id).price, Auction(auction\_id).player != player\_id    2. 如果是暗拍, 确认Bids表中没有同时满足player\_id和auction\_id的数据   4.1.明拍: 向Auction(auction\_id).player的gold\_frozen中减去Auction(auction\_id).price, 向player\_id的gold\_frozen中增加price. 更新Auctions(auction\_id)中的price和player  4.2. 暗拍: 向player\_id的gold\_frozen中增加price. 如果price> Auction(auction\_id).price, 更新Auctions(auction\_id)中的price和player  5. 向Bids表中添加新数据, Aucions的count\_bid++  6. 修改Aucion的time\_latest  7. 返回auction\_id  8． 返回BID\_SUCCESS/ BID\_FAILED\_NOTENOUGHGOLD/ BID\_FAILED\_PRICETOOLOW/ BID\_FAILED\_HASBIDDED/ BID\_ERROR |

|  |  |
| --- | --- |
| 方法名 | enterAuctionCenter |
| 说明 | 初次进入拍卖行，初始化数据  拍卖行未到人数上限  更新Player和AuctionCenter数据 |
| 参数 | player\_id, AuctionCenter\_id |
| 返回值 | statuscode |
| 操作 | 1. 确认AuctionCenter\_id存在 2. 确认AuctionCenter(AuctionCenter\_id).count< AuctionCenter(AuctionCenter\_id).max\_ player 3. 更新Player(player\_id).AuctionCenter, AuctionCenter(AuctionCenter\_id).count\_player++ 4. 根据初始数量INITIAL\_CARD, 调用supply方法发牌 5. 返回ENTERAUCTIONCENTER\_SUCCESS / ENTERAUCTIONCENTER\_ERROR |

|  |  |
| --- | --- |
| 方法名 | exitAuctionCenter |
| 说明 | 离开拍卖行，清除数据 |
| 参数 | player\_id, AuctionCenter\_id |
| 返回值 | statuscode |
| 操作 | 1. 确认player\_id和AuctionCenter\_id 2. 确认Player(player\_id).on\_sale==false, Bids表中无player.id的数据 3. 更新Player(player\_id).AuctionCenter, AuctionCenter(AuctionCenter\_id).count\_player-- 4. 清空Player\_Cards中所有player\_id相关项 5. 返回EXITAUCTIONCENTER\_SUCCESS/ EXITAUCTIONCENTER\_FAILED\_ONSALE/ EXITAUCTIONCENTER\_ERROR |

|  |  |
| --- | --- |
| 方法名 | getPlayer |
| 说明 | 获得player信息 |
| 参数 | player\_id |
| 返回值 | statuscode, Player |
| 操作 | 1. 确认player\_id存在 2. 返回gold, exp, AuctionCenter\_id。 3. 返回GETPLAYER\_SUCCESS/ GETPLAYER\_ERROR |

|  |  |
| --- | --- |
| 方法名 | getAuctionCenter |
| 说明 | 获得AuctionCenter信息 |
| 参数 | AuctionCenter\_id |
| 返回值 | statuscode, auctioncenter, pricelist |
| 操作 | 1. 确认AuctionCenter\_id存在 2. 从RepurchasePrice表中查询AuctionCenter\_id相关条目 3. 返回statuscode, AuctionCenter, RepurchasePrice的链表 4. 返回GETAUCTIONCENTER\_SUCCESS/ GETAUCTIONCENTER\_ERROR |

|  |  |
| --- | --- |
| 方法名 | getAuction |
| 说明 | 获得auction信息 |
| 参数 | auction\_id |
| 返回值 | statuscode, Auction |
| 操作 | 1. 确认auction\_id存在 2. 返回card, time\_start, time\_last, price, bid\_count 3. 返回GETAUCTION\_SUCCESS/ GETAUCTION\_ERROR |

|  |  |
| --- | --- |
| 方法名 | getAuctionList（只get auction\_id？） |
| 说明 | 获得正在进行的拍卖列表 |
| 参数 | AuctionCenter \_id |
| 返回值 | statuscode, auction\_list |
| 操作 | 1. 确认AuctionCenter \_id存在 2. 返回aution\_list 3. 返回GETAUCTIONLIST\_SUCCESS/ GETAUCTIONLIST\_ERROR |

|  |  |
| --- | --- |
| 方法名 | getBid |
| 说明 | 获得某用户参与竞价的auction信息 |
| 参数 | player\_id |
| 返回值 | statuscode, auction\_list |
| 操作 | 1. 确认player\_id存在 2. 从Bids表中查找相关auction数据 3. 返回auction\_list 4. 返回GETBID\_SUCCESS/ GETBID\_ERROR |

|  |  |
| --- | --- |
| 方法名 | ackNewCard |
| 说明 | 确认收到新物品 |
| 参数 | player\_id |
| 返回值 | statuscode |
| 操作 | 1. 确认player\_id存在 2. 将player\_card中相关数据is\_new设为false |

服务器定时操作

|  |  |
| --- | --- |
| 方法名 | deal |
| 说明 | 拍卖成交  拍卖到时间后，根据最高出价人信息更新数据库 |
| 参数 |  |
| 返回值 |  |
| 操作 | 1. 遍历Auctions, 发现到时限的拍卖 2. 如果没有bidder，将卡还给owner 3. 根据bidder信息，向player\_card表中添加新数据，增加买家的exp, 从gold中扣除price,减去gold\_frozen 4. 根据owner信息，增加对应卖家的gold,exp 5. 如果是暗拍，根据bids表中信息，退还其余买家的gold\_frozen 6. Prices表中对应count++ 7. 清除auction数据及相关bid信息 |

|  |  |
| --- | --- |
| 方法名 | repurchase |
| 说明 | 每天定时回购  结束拍卖，定价，回购，补牌 |
| 参数 |  |
| 返回值 |  |
| 操作 | 1. 结束所有未完结拍卖 2. 根据prices中的count值进行排序并定价 3. 根据player\_card表中的card\_bought属性，增加用户gold,exp 4. 清除card\_bought相关数据 5. 根据时间段，调用supply补牌 |

服务器端其他

|  |  |
| --- | --- |
| 方法名 | supply |
| 说明 | 随机补n张牌 |
| 参数 | player\_id，num |
| 返回值 |  |
| 操作 | 1. 确认player\_id的存在 2. Num = num>max\_card-count\_card ? max\_card-count\_card : num 3. 随机构造出num张牌，is\_new=true，添加至player\_card表中 |

V2补充

|  |  |
| --- | --- |
| 方法名 | getCardList（只get card\_id？） |
| 说明 | 获得player信息 |
| 参数 | player\_id |
| 返回值 | statuscode, card\_list |
| 操作 | 1. 确认player\_id存在 2. 从Player\_Card表中取得该player相关card信息 3. 返回card\_list 4. 返回GETCARDLIST\_SUCCESS/ GETCARDLIST\_ERROR |

|  |  |
| --- | --- |
| 方法名 | getAuctionCenterList |
| 说明 | 获得所有的AuctionCenter列表 |
| 参数 |  |
| 返回值 | statuscode, auctioncenterlist |
| 操作 | 1. 从AuctionCenter表中查询所有条目 2. 返回statuscode, AuctionCenter的链表 3. 返回GETAUCTIONCENTERLIST\_SUCCESS/ GETAUCTIONCENTERLIST\_ERROR |

补充（2010.03.02）

|  |  |
| --- | --- |
| 方法名 | getSystemTime |
| 说明 | 获得系统当前时间 |
| 参数 |  |
| 返回值 | time |
| 操作 | 返回datetime.now  返回GETTIME\_SUCCESS/ GETTIME\_ERROR |

补充（2010.03.07）

|  |  |
| --- | --- |
| 方法名 | logPlayerMsg |
| 说明 | 记录player相关消息（server内部方法） |
| 参数 | player\_id, message |
| 返回值 |  |
| 操作 | 向PlayerLog表中记录数据 |
| 调用场景 | 进入/离开拍卖行  拍卖  竞拍  拍卖成功  流拍  竞拍成功  竞拍失败  被回购 |

|  |  |
| --- | --- |
| 方法名 | getPlayerLog |
| 说明 | 获得player日志 |
| 参数 | player\_id |
| 返回值 | statuscode, loglist |
| 操作 | 1. 从PlayerLog表中查询所有条目 2. 返回statuscode, log的链表   3.返回GETPLAYERLOG\_SUCCESS/ GETPLAYERLOG\_ERROR |